UNIVERSITÉ DE STRASBOURG

ÉCOLE DOCTORALE Mathématiques,

Sciences de l'Information et de l'Ingénieur

Institut de Recherche Mathématique Avancée, UMR 7501

THÈSE présentée par :

Christophe STEINER

soutenue le : **11 décembre 2014**

pour obtenir le grade de : **Docteur de l'Université de Strasbourg**

Discipline / Spécialité : MATHÉMATIQUES APPLIQUÉES

Résolution numérique de l'opérateur de gyromoyenne, schémas d'advection et couplage. Applications à l'équation de Vlasov.

THÈSE dirigée par :

Mr. MEHRENBERGER Michel	Maître de Conférences HDR, Université de Strasbourg
Mr. CROUSEILLES Nicolas	Chargé de Recherche HDR, INRIA Rennes Bretagne Atlantique

RAPPORTEURS :

Mr. BESSE Nicolas	Maître de Conférences HDR, Université de Lorraine
Mr. DIMARCO Giacomo	Associate Professor, Université de Ferrara

EXAMINATEURS :

Mr. DESPRES Bruno	Professeur, Université Paris 6 UPMC
Mr. HELLUY Philippe	Professeur, Université de Strasbourg

Remerciements

Mes premiers remerciements vont tout naturellement à mes directeurs de thèse, Michel Mehrenberger et Nicolas Crouseilles, qui m'ont encadré durant ces trois années avec beaucoup de disponibilité et une grande gentillesse. Je remercie Michel pour sa bienveillance, son enthousiasme et la patience dont il a fait preuve à mon égard sans jamais compter ses heures. Ce travail n'aurait pas vu le jour sans les nombreuses explications et indications qu'il a pu me fournir. Je remercie Nicolas pour m'avoir fait partager ses larges connaissances, pour sa détermination et pour sa sympathie. Il aura, malgré l'éloignement géographique, toujours été présent et a suivi de près mon travail.

Je souhaite exprimer ma gratitude à Nicolas Besse et Giacomo Dimarco qui m'ont fait l'honneur d'accepter de rapporter ce manuscrit. Je les remercie d'avoir utilisé de leur temps précieux à la relecture attentive de cette thèse.

Je remercie Bruno Després et Philippe Helluy d'avoir accepté d'être mes examinateurs ainsi que Mihaï Bostan et Giovanni Manfredi d'avoir accepté de faire partie du jury.

Cette thèse s'est déroulée à l'IRMA au sein de l'équipe Modélisation et Contrôle. Je voudrais remercier tous les membres, actuels et anciens pour leurs conseils et leur bienveillance. Un grand merci à Bopeng, Christophe, Edwin, Emmanuel, Laurent, Marcela, Mickaël et Sever. Je voudrais exprimer plus particulièrement ma gratitude à Vilmos qui m'a permis de débuter cette thèse et à Pierre qui a toujours été d'un grand secours dans la résolution de mes problèmes informatiques. Je suis très reconnaissant à l'ensemble du personnel administratif d'INRIA Nancy Grand-Est, de l'UFR et de l'IRMA, les documentalistes, les secrétaires, l'équipe informatique, le personnel technique... qui nous facilitent les tâches administratives au quotidien.

Ce travail est le fruit de nombreuses collaborations. Je remercie Daniel Bouche de m'avoir fait partager ses nombreuses connaissances sur les équations équivalentes et de m'avoir permis de présenter mes travaux à l'ENS. Je souhaiterai ensuite remercier l'équipe du CEA et en particulier Virginie Grandgirard, Guillaume Latu, Fabien Rozar et Thomas Cartier-Michaud pour les nombreuses discussions sur la théorie gyrocinétique ainsi que Éric Sonnendrücker pour avoir partagé de son temps précieux, sa grande expérience et son savoir.

L'enseignement a pris une part importante pendant mes trois années de thèse à Strasbourg. Lors de chaque année de monitorat, j'ai eu la chance de m'intégrer dans des équipes pédagogiques très sympathiques et je voudrais ici les en remercier. Je souhaite exprimer ma gratitude aux professeurs m'ayant donné l'envie de poursuivre des études en mathématiques, en particulier M. Barthel et M. Pister. Je remercie également Claudine

Mitschi pour sa bienveillance sans faille à l'égard des magistériens.

Le plaisir que l'on prend à effectuer un travail tient également pour une grande part à l'environnement dans lequel on évolue. Je remercie les autres doctorants et particulièrement Ambroise, Arnaud, Camille, Guillaume, Mathieu et Olivier pour leur bonne humeur et leur amitié. Je remercie Pierre d'avoir partagé avec moi des compétitions de course à pied et quelques centaines de tours de l'Orangerie. Merci à tous ceux ayant participé au CEMRACS 2012 et notamment Benoît, Céline, Elisa, Jonathan, Lucas, Ranine et Tony avec qui j'ai passé un très agréable été. Je remercie mes co-bureaux, Anaïs et Nhung, pour tous les moments de détente ainsi que ceux avec qui ce chemin a commencé avant le début de la thèse : Adrien, Arnaud, Boris, Claire, Maurice, Médéric et Philippe. Je tiens à remercier Hichem qui aura été un parfait binôme lors de la préparation à l'agrégation et Gilles pour son soutien inconditionnel. Un grand merci à Daniel, alias Jean Bombeur, alias Hans Aplast, alias Pierre Serre qui m'a supporté comme colocataire pendant 5 années.

Je souhaite enfin remercier ma famille à qui je dois énormément. Merci infiniment à ma sœur Élodie et mes parents Dominique et Doris pour avoir toujours cru en moi et veillé à chaque instant à ce que je ne manque de rien. Enfin, je remercie de tout cœur Delphine pour ses encouragements et tant d'autres raisons.

Résumé

Cette thèse propose et analyse des méthodes numériques pour la résolution de l'équation de Vlasov. Cette équation modélise l'évolution d'une espèce de particules chargées sous l'effet d'un champ électromagnétique. La première partie est consacrée à une analyse mathématique de schémas semi-Lagrangiens résolvant l'équation de transport linéaire qui constituent la brique de base des méthodes de splitting directionnel. Des méthodes de résolution de l'équation de Vlasov couplée à l'équation de Poisson, dans le cas où uniquement le champ électrique est considéré, sont optimisées dans la seconde partie. Il s'agit d'optimisation en temps de calcul par l'utilisation de cartes graphiques (GPU) et l'utilisation d'un maillage non homogène. Dans la troisième et dernière partie, nous étudierons une méthode numérique de calcul de l'opérateur de gyromoyenne intervenant dans la théorie gyrocinétique que nous appliquerons à l'équation de quasi-neutralité.

Chapitre 1

Ce premier chapitre présente le contexte physique du confinement magnétique du plasma ainsi que sa modélisation mathématique à travers les systèmes de Vlasov-Maxwell ou de Vlasov-Poisson. Nous donnerons en particulier certaines propriétés de ces systèmes. Un court résumé des méthodes de résolution classiquement utilisées sera présenté en s'attardant sur les méthodes semi-Lagrangiennes qui constituent le cadre d'étude de cette thèse. Finalement, nous donnerons les grands défis dans la résolution numérique de ces modèles.

Méthodes d'advection

Chapitre 2

Pour les EDP linéaires résolus par des maillages à pas constants, l'analyse de Fourier donne la solution exacte du schéma. Cette méthode n'est plus applicable pour les équations non linéaires ou les équations d'advection à vitesse variable. Dans ce second chapitre, nous présentons la méthode de l'équation équivalente qui consiste à déterminer l'EDP équivalente à l'équation aux différences. Cette méthode a été introduite pour d'une part traiter les cas non linéaires et d'autre part fournir des solutions explicites (et non plus sous représentation intégrale) pour les EDP linéaires. L'étude de ces équations équivalentes permet en particulier d'accéder aux propriétés dispersives et dissipatives des schémas et de les comparer entre elles. Dans un premier temps, nous établirons les équations équivalentes de schémas semi-Lagrangiens résolvant l'équation d'advection constante puis nous étudions un schéma de type Lagrange+Projection utilisé classiquement pour résoudre les équations de l'hydro-

dynamique.

Chapitre 3

Nous discutons, dans ce chapitre, de la propriété de superconvengence pour le schéma Galerkin Discontinu Semi-Lagrangien (SLDG). Un point clé, dans les applications à Vlasov-Maxwell/Poisson, est d'utiliser les splittings directionnels qui conduisent à une succession de problèmes d'advection constante et le schéma a l'avantage de ne pas être restreint par une condition CFL. Dans le cadre de l'équation d'advection linéaire avec des conditions aux bords périodiques, nous montrons une propriété de superconvergence pour le schéma Galerkin Discontinu Semi-Lagrangien pour les petits degrés. Cette propriété sera vérifiée numériquement et formellement. Nous donnerons des pistes pour l'établissement d'une preuve valable pour un degré quelconque.

Méthodes pour l'équation de Vlasov-Poisson

Chapitre 4

Le contexte de l'équation de Vlasov-Poisson permet une opération de splitting licite. Cependant, dans certaines situations, cette procédure n'est pas appropriée et peut conduire à des instabilités numériques. L'objectif principal de ce chapitre est de rechercher des versions non splittées de schémas de volumes finis. Deux stratégies de type volumes finis sont ici étudiées pour l'approximation de l'équation de Vlasov-Poisson $1D \times 1D$. Une analyse de stabilité pour ces schémas est effectuée dans le cas de l'advection linéaire unidimensionnelle. Plusieurs liens sont faits entre les méthodes volumes finis et semi-Lagrangiennes. Enfin, les méthodes sont comparées sur deux cas tests académiques de la physique des plasmas.

Chapitre 5

Les ondes KEEN sont un cas test difficile pour les solveurs numériques de l'équation de Vlasov-Poisson puisqu'elles comprennent des états cinétiques multi-harmoniques, auto-organisés et fortement non stationnaires. Afin d'obtenir la haute résolution nécessaire à la résolution numérique de ce cas test en un temps raisonnable, nous avons étudié deux stratégies : l'accélération du code sur carte graphique GPU (qui sera l'objet de ce chapitre) et l'utilisation d'un maillage non uniforme en vitesse (qui sera l'objet du chapitre suivant). Ce chapitre présente une accélération du code de résolution de l'équation de Vlasov-Poisson sur carte graphique GPU en utilisant une méthode semi-Lagrangienne. Les conditions périodiques nous permettent de formuler l'interpolation sous forme d'une matrice circulante diagonalisable dans la base de Fourier. Les simulations bénéficient alors de l'importante accélération obtenue par la FFT sur GPU. Les performances des codes sur CPU et GPU sont comparées sur plusieurs cas tests.

Chapitre 6

Le cas test des ondes KEEN nécessite une haute résolution dans la région de l'espace des phases autour d'une vitesse caractéristique. Après des travaux antérieurs utilisant des splines cubiques non uniformes, nous présentons ici un schéma Galerkin Discontinu

Semi-Lagrangien (SLDG) qui résout l'équation d'advection constante unidimensionnelle sur un maillage non structuré. Nous montrerons que dans le cas test des ondes KEEN, très localisées en vitesse, nous obtenons des résultats similaires à ceux du chapitre précédent en diminuant le nombre de points en vitesse.

Modèle gyrocinétique

Chapitre 7

Dans ce chapitre, nous présentons un opérateur d'interpolation de type Hermite. Cette méthode d'interpolation a l'avantage d'être locale, de faible complexité et souple par le choix du degré arbitraire de la reconstruction des dérivées. Nous avons utilisé cet interpolateur dans le cadre de la résolution numérique du modèle Drift-Kinetic 4D en géométrie SLAB. Nous montrons que son influence est radicalement différente suivant que la reconstruction des dérivées est centrée ou décentrée et nous ferons des comparaisons avec l'interpolation par splines cubiques.

Chapitre 8

Nous nous intéressons, dans ce chapitre, à l'approximation numérique des opérateurs de gyromoyenne intervenant en physique des plasmas afin de prendre en compte les effets des rayons de Larmor finis. Ce travail, initié en géométrie cartésienne, est prolongé ici en géométrie polaire. Une méthode directe est proposée dans l'espace des configurations qui consiste à intégrer sur le cercle de giration en utilisant des opérateurs d'interpolation (Hermite ou splines cubiques). Des comparaisons numériques avec la méthode standard basée sur l'approximation de Padé sont effectuées : (i) sur des solutions analytiques, (ii) sur le modèle drift-kinetic 4D avec un rayon de Larmor fixé (iii) sur le classique cas test linéaire DIII-D. Nous montrons que, dans le cas linéaire, les différences avec Padé sont importantes en prenant une géométrie SLAB et un rayon relativement grand. En outre, l'introduction de l'opération de gyromoyenne tend à diminuer le taux d'instabilité et cela est amplifié en considérant l'opérateur de gyromoyenne direct au lieu de l'approximation de Padé ce qui sera validé par l'étude de la relation de dispersion.

Chapitre 9

Dans le cadre des équations gyrocinétiques, nous développons un solveur pour l'équation de quasi-neutralité basé sur la méthode de calcul de la gyromoyenne en géométrie polaire décrite dans le chapitre 8. Nous comparerons ce solveur à la méthode classique par Padé sur des cas tests analytiques et montrerons que ce nouveau solveur est avantageux pour de hauts modes en theta. Nous considérerons également un modèle simplifié de l'équation de quasi-neutralité avec un μ (au lieu d'une intégrale en μ) sur lequel nous comparerons le nouveau solveur avec le solveur par Padé dans le cadre de simulations gyrocinétiques. Nous montrerons en particulier le taux d'instabilité pour le solveur par Padé sera le plus faible.

Publications et communications

Les résultats de cette thèse ont fait l'objet de publications ainsi que de posters.

- Les résultats du chapitre 2 portant sur les schémas semi-Lagrangiens conservatifs sont tirés de mon stage de M2 :

  C. Steiner, *Equivalent equations for the linear advection equation*, Stage M2, 2011.

  La partie concernant le schéma Lagrange+Projection est une étude répondant à un contrat CEA *Determination of numerical diffusion in a Lagrange-Projection type scheme* (2011).

- Le chapitre 4 a partiellement été présenté lors du CEMRACS 2011 et est la traduction du proccedings qui a été publié dans les actes de celui-ci :

  N. Crouseilles, P. Glanc, M. Mehrenberger & C. Steiner, *Finite Volume Schemes for Vlasov*, ESAIM 2011 Proceedings, 2012.

- Le chapitre 5 a partiellement été présenté lors du CEMRACS 2012 et est la traduction du proccedings qui a été publié dans les actes de celui-ci :

  M. Mehrenberger, C. Steiner, L. Marradi, N. Crouseilles, E. Sonnendrucker & B. Afeyan, *Vlasov on GPU*, ESAIM 2012 Proceedings, 2014.

- Les résultats du chapitre 6 ont fait l'objet d'un poster lors du congrès *VLASOVIA 2013*.

- Le chapitre 8 a partiellement été présenté sous forme d'un poster au congrès *CANUM 2012* puis les résultats ultérieurs ont été intégrés dans l'article suivant, dont le chapitre 8 est la traduction :

  C. Steiner, M. Mehrenberger, N. Crouseilles, V. Grandgirard, G. Latu & F. Rozar *Gyroaverage operator for a polar mesh*, accepté à Eur. Phys. J. D (2014).

Table des matières

Chapitre 1

Introduction à la physique des plasmas

1.1 Confinement magnétique

1.1.1 Les plasmas

Le plasma est, après l'état solide, liquide et gazeux, le $4^{ème}$ état de la matière. Il apparait lorsqu'un gaz est placé à très haute température ce qui provoque le détachement d'électrons des atomes. On a alors affaire à un gaz ionisé. La plasma est l'état largement prédominant dans l'univers puisque celui-ci regroupe plus de 99,9% de la matière visible ; en particulier, il constitue les étoiles. Sur Terre, le plasma est présent dans les aurores boréales et est utilisé dans les tubes néons et les écrans plasmas.

1.1.2 Réaction de fusion

La production d'électricité est un enjeu majeur pour le futur du à l'appauvrissement des ressources fossiles et à l'accroissement des besoins mondiaux. L'énergie peut être produite à travers les réactions nucléaires. Une réaction nucléaire est la transformation de noyaux atomiques ; il en existe de deux types : la fission et la fusion.

- La fission nucléaire se produit lorsque le noyau d'un atome lourd se scinde en des nucléides plus légers, généralement sous l'effet de la collision avec un neutron. Cette fragmentation libère alors de l'énergie. Ce principe est utilisé dans les centrales nucléaires actuelles.
- La fusion nucléaire (ou thermonucléaire) se produit, quant à elle, lorsque deux noyaux entrent en collision pour n'en donner plus qu'un plus lourd. Contrairement à la fission nucléaire, son application industrielle est encore à l'état de recherche. La réaction de fusion la plus accessible est celle faisant intervenir le deutérium et le tritium qui sont deux isotopes de l'hydrogène (voir Fig. **??**).

Une difficulté de la faisabilité technique de la fusion thermonucléaire est que les noyaux doivent être portés à une très haute température (plus d'un million de degrés) pour que la réaction de fusion se produise. A de telles températures, les électrons sont détachés des atomes et on est ainsi en présence d'un plasma. Il existe deux approches permettant de confiner les réactions de fusion : le confinement inertiel et le confinement magnétique. Le

confinement inertiel consiste à projeter un faisceau laser sur une capsule de deutérium et de tritium. Le confinement magnétique, qui est le cadre d'étude de cette thèse, utilise un champ magnétique avec une densité plus faible mais sur un temps plus long. La plasma est alors confiné dans une chambre toroïdale ou Tokamak (Fig. 1.1). Cette chambre est entourée d'aimants produisant le champ magnétique permettant le confinement, ce qui évite que le plasma entre en contact avec les parois du Tokamak et l'endommage.

Le projet ITER (`www.iter.org`) est un partenariat international, comprenant l'Union Européenne, l'Inde, le Japon, la Chine, la Russie, la Corée du Sud, les Etats-Unis et la Suisse, qui a pour objectif de vérifier la faisabilité scientifique et technique de la production d'électricité en utilisant le principe de la fusion thermonucléaire par confinement magnétique. Ce projet a été signé le 21 novembre 2006 à Paris.

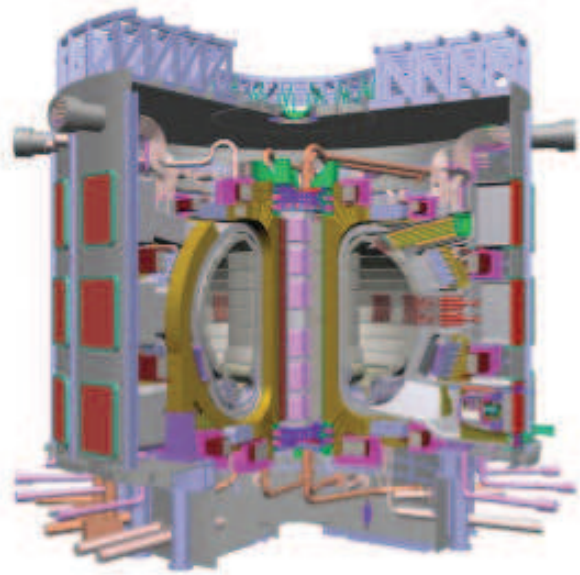


FIGURE 1.1 – Vue d'artiste de ITER.

1.2 Modélisation des plasmas

Le plasma a été modélisé par une hiérarchie de modèles : le modèle à N corps, les modèles cinétiques et les modèles fluides. Le modèle à N corps, qui consiste à modéliser l'interaction de chacune des particules, est le modèle le plus précis mais n'est pas accessible au niveau numérique à cause du nombre élevé de particules. Les modèles cinétiques donnent, quant à eux, une description statistique de la répartition des particules dans le plasma à travers la fonction de distribution dans l'espace des phases. Les modèles fluides réduisent les modèles précédents en considérant l'évolution des quantités macroscopiques telles que la densité ou la vitesse moyenne qui sont des moments de la fonction de distribution.

1.2.1 Le modèle à N corps

A niveau microscopique, le plasma peut être décomposé en une collection de particules interagissant les unes avec les autres. Chaque particule obéit à la loi de Newton :

$$\frac{d\mathbf{p}}{dt} = \sum F_{ext}$$

où $\mathbf{p}$ est la quantité de mouvement de la particule et $\sum F_{ext}$ est la somme des forces extérieures appliquées sur la particule. La quantité de mouvement peut s'exprimer par la relation

$$\mathbf{p} = m\gamma\mathbf{v}$$

où m est la masse de la particule, $\mathbf{v}$ désigne sa vitesse et $\gamma = \left(1 - \frac{|\mathbf{v}|^2}{c^2}\right)^{-1/2}$ est le facteur de Lorentz dans lequel intervient la vitesse de la lumière dans le vide c. Les forces extérieures se réduisent à la force de Lorentz

$$\sum F_{ext} = \sum_j q(\mathbf{E}_j + \mathbf{v}_i \times \mathbf{B}_j)$$

où q est la charge d'une particule, $q\mathbf{E}_j$ et $q\mathbf{v}_i \times \mathbf{B}_j$ sont les forces électriques et magnétiques produites soit par les sources extérieures, soit par les autres particules. L'évolution de chaque particule i est alors décrite par le système d'équations

$$\begin{aligned} \frac{d(m\gamma_i\mathbf{v}_i)}{dt} &= \sum_j q(\mathbf{E}_j + \mathbf{v}_i \times \mathbf{B}_j) \\ \frac{d\mathbf{x}_i}{dt} &= \mathbf{v}_i. \end{aligned}$$

Le plasma est composé d'énormément de particules (plus de 10^{10}) et la simulation numérique de l'interaction de toutes les particules en suivant ce modèle est hors de portée. Nous allons nous tourner alors vers des modèles moins précis mais plus raisonnables d'un point de vue numérique que sont les modèles cinétiques et fluides.

1.2.2 Les modèles cinétiques

Le modèle cinétique décrit de manière statistique la répartition des différentes espèces de particules dans le plasma à travers leur fonction de distribution $f_s(\mathbf{x}, \mathbf{v}, t)$ associée à chaque espèce qui dépend de la position $\mathbf{x} \in \mathbb{R}^3$, de la vitesse $\mathbf{v} \in \mathbb{R}^3$ et du temps t. Plus précisément, $f_s(\mathbf{x}, \mathbf{v}, t)d\mathbf{x}d\mathbf{v}$ représente la probabilité de trouver des particules de l'espèce s dans l'élément de volume $d\mathbf{x}d\mathbf{v}$ au temps t et au point $(\mathbf{x}, \mathbf{v})$. Dans la suite, nous ne considérons plus que les électrons et nous noterons f leur fonction de distribution.

En partant de l'équation de transport des particules

$$\frac{\partial f}{\partial t} + \mathbf{v} \cdot \nabla_{\mathbf{x}} f = 0$$

et en ajoutant le terme dérivant du modèle à N corps par une étude de physique statistique, nous obtenons l'équation de Vlasov

$$\frac{\partial f}{\partial t} + \mathbf{v} \cdot \nabla_{\mathbf{x}} f + \frac{q}{m}(\mathbf{E} + \mathbf{v} \times \mathbf{B}) \cdot \nabla_{\mathbf{v}} f = 0.$$

1.2.3 Les modèles fluides

Les modèles fluides donnent une description macroscopique en étudiant les quantités physiques découlant de la fonction de distribution f telle que la densité ρ :

$$\rho(\mathbf{x}, t) = \int_{\mathbb{R}^3} f(\mathbf{x}, \mathbf{v}, t) d\mathbf{v}$$

la vitesse moyenne $\mathbf{u}$:

$$\rho(\mathbf{x}, t)\mathbf{u}(\mathbf{x}, t) = \int_{\mathbb{R}^3} \mathbf{v} f(\mathbf{x}, \mathbf{v}, t) d\mathbf{v}$$

ou la température T :

$$\rho(\mathbf{x}, t)T(\mathbf{x}, t) = \frac{m}{3} \int_{\mathbb{R}^3} |\mathbf{v} - \mathbf{u}(\mathbf{x}, t)|^2 f(\mathbf{x}, \mathbf{v}, t) d\mathbf{v}$$

qui sont les moments d'ordre 0, 1 et 2 de la fonction de distribution en $\mathbf{v}$ et ne dépendent donc que de la position $\mathbf{x}$ et du temps t. Afin de trouver les équations qui relient ces quantités, nous pouvons considérer les premiers moments en vitesse de l'équation de Vlasov (voir [5]). Ne dépendant plus de $\mathbf{v}$, ces systèmes sont moins coûteux à résoudre numériquement mais sont moins généraux que les modèles cinétiques. Les modèles fluides constituent une bonne approximation lorsque, en temps long, le système s'approche de l'état d'équilibre thermodynamique sous l'effet des collisions. En effet, la distribution des particules en vitesse est alors une gaussienne lorsque l'on est proche de cet état d'équilibre.

1.3 Equations de Vlasov, Maxwell et Poisson

1.3.1 Equation de Vlasov

Le cadre d'étude de cette thèse est le modèle cinétique qui donne une description statistique de la répartition des particules à travers la fonction de distribution $f(\mathbf{x}, \mathbf{v}, t)$ dans l'espace des phases. Comme nous l'avons vu précédemment, cette fonction est solution de l'équation de Vlasov :

$$\frac{\partial f}{\partial t} + \mathbf{v} \cdot \nabla_{\mathbf{x}} f + \frac{q}{m}(\mathbf{E} + \mathbf{v} \times \mathbf{B}) \cdot \nabla_{\mathbf{v}} f = 0$$

où m (resp. q) est la masse (resp. la charge) des particules. Cette équation a la forme d'une équation de transport ; elle est non linéaire puisque les termes de champ $\mathbf{E}$ et $\mathbf{B}$ dépendent de f à travers les équations de Maxwell ou de Poisson.

1.3.2 Equations de Maxwell

Lorsque l'on prend en compte le champ électromagnétique auto-consistant généré par les particules, l'équation de Vlasov est couplée aux équations de Maxwell :

$$\begin{aligned} -\frac{1}{c^2}\frac{\partial \mathbf{E}}{\partial t} + \nabla \times \mathbf{B} &= \mu_0 \mathbf{J} \\ \frac{\partial \mathbf{B}}{\partial t} + \nabla \times \mathbf{E} &= 0 \\ \nabla \cdot \mathbf{E} &= \frac{\rho}{\varepsilon_0} \\ \nabla \cdot \mathbf{B} &= 0 \end{aligned}$$

où μ_0 la perméabilité magnétique du vide, ε_0 est la permittivité diélectrique du vide, ρ la densité de charge définie par

$$\rho(\mathbf{x}, t) = \int_{\mathbb{R}^3} f(\mathbf{x}, \mathbf{v}, t) d\mathbf{v}$$

et $\mathbf{J}$ la densité de courant définie par

$$\mathbf{J}(\mathbf{x}, t) = \int_{\mathbb{R}^3} f(\mathbf{x}, \mathbf{v}, t) \mathbf{v} d\mathbf{v}.$$

1.3.3 Equation de Poisson

Afin de simplifier le modèle précédent, nous supposons que les champs électrique et magnétique ne dépendent plus (ou très peu) du temps. Les équations de Maxwell adimensionnées se réécrivent alors :

$$\begin{aligned} \nabla \times \mathbf{B} &= \mathbf{J} \\ \nabla \times \mathbf{E} &= 0 \\ \nabla \cdot \mathbf{E} &= \rho \\ \nabla \cdot \mathbf{B} &= 0 \end{aligned}$$

Nous négligeons le champ magnétique qui est supposé faible. L'équation $\nabla \times \mathbf{E} = 0$ implique que le champ $\mathbf{E}$ dérive d'un potentiel : $\mathbf{E} = -\nabla\phi$ qui vaut $-\Delta\phi = \rho$ d'après la relation $\nabla \cdot \mathbf{E} = \rho$. Il s'agit de l'équation de Poisson. Le système de Vlasov-Poisson s'écrit alors

$$\begin{cases} \frac{\partial f}{\partial t} + \mathbf{v} \cdot \nabla_x f - \mathbf{E} \cdot \nabla_v f = 0 \\ \\ -\Delta\phi = 1 - \rho, \qquad \mathbf{E} = -\nabla\phi \end{cases}$$

avec

$$\rho(\mathbf{x}, t) = \int_{\mathbb{R}^3} f(\mathbf{x}, \mathbf{v}, t) d\mathbf{v}.$$

1.3.4 Propriétés mathématiques

Le système de Vlasov-Poisson a la propriété de conserver plusieurs quantités physiques au cours du temps.

Proposition 1.3.1 (Principe du maximum). *Soit $f_0(x, v)$ la fonction de distribution initiale du système de Vlasov-Poisson et $f(x, v, t)$ sa solution alors*

$$0 \leq f(x, v, t) \leq \max_{\boldsymbol{x},\boldsymbol{v}} f_0(\boldsymbol{x}, \boldsymbol{v}).$$

Proposition 1.3.2 (Conservation). *Les quantités suivantes sont conservées au cours du temps :*

— *L'énergie cinétique :*

$$\mathcal{E}_k(t) = \frac{1}{2} \int_{\mathbb{R}^6} \boldsymbol{v}^2 f(\mathbf{x}, \mathbf{v}, t) d\mathbf{x} d\mathbf{v}$$

— *L'énergie électrique :*

$$\mathcal{E}_e(t) = \frac{1}{2} \int_{\mathbb{R}^3} E^2(\mathbf{x}, t) d\mathbf{x}$$

— *L'énergie totale*

$$\mathcal{E}(t) = \mathcal{E}_k(t) + \mathcal{E}_e(t)$$

— *Pour tout entier $p \geq 1$, la norme L^p :*

$$\|f\|_{L^p}^p(t) = \int_{\mathbb{R}^6} (f(\mathbf{x}, \mathbf{v}, t))^p d\mathbf{x} d\mathbf{v}$$

et en particulier la masse :

$$\|f\|_{L^1}(t) = \int_{\mathbb{R}^6} f(\mathbf{x}, \mathbf{v}, t) d\mathbf{x} d\mathbf{v}$$

— *Le moment*

$$\int_{\mathbb{R}^6} \boldsymbol{v} f(\mathbf{x}, \mathbf{v}, t) d\mathbf{x} d\mathbf{v}.$$

Lors du développement de méthodes numériques pour la résolution de l'équation de Vlasov-Poisson, un intérêt spécial doit être apporté à la conservation de ces quantités physiques. Bien que toutes ces quantités ne puissent pas être conservées de manière exacte, leur évolution en temps est un outil précieux afin de valider une simulation.

1.4 Résolution numérique

1.4.1 Classifications des méthodes

L'équation de Vlasov-Maxwell (ou de Vlasov-Poisson) étant non linéaire, il est difficile de trouver des solutions analytiques. Plusieurs familles de méthodes numériques ont été proposées afin de résoudre numériquement ces systèmes. Il existe principalement deux types de méthodes : les méthodes se basant sur la discrétisation de la fonction de distribution dans l'espace des phases (*méthodes Eulériennes* ou *semi-Lagrangiennes*) et les *méthodes PIC* (Particule In Cell).

Les méthodes PIC consistent à suivre le déplacement d'une collection de N macro-particules aux points $(\mathbf{x}_k(t), \mathbf{v}_k(t))$ et de poids ω_k qui discrétisent la fonction de distribution :

$$f_N(\mathbf{x}, \mathbf{v}, t) = \sum_{k=1}^{N} \omega_k \delta(\mathbf{x} - \mathbf{x}_k(t)) \delta(\mathbf{v} - \mathbf{v}_k(t)).$$

Les macro-particules sont alors advectées le long des caractéristiques de l'équation de Vlasov. Les méthodes PIC n'ont pas besoin de grille en vitesse ce qui constitue un avantage au niveau de la mémoire et du coût de calcul ; ainsi, cette famille de méthodes est quasiment la seule approche permettant de faire de la "vraie" physique. Cependant, une faible résolution est observée dans les régions où il y a peu de particules. La majorité des codes actuels sont basés sur ces méthodes.

Les méthodes purement Eulériennes (volumes finis et différences finies) ont besoin d'une grille de l'espace des phases (coût mémoire important en 6 dimensions) mais permettent d'avoir une résolution uniforme sur tout l'espace des phases (sous condition de régularité...).

Les méthodes semi-Lagrangiennes, qui seront étudiées dans cette thèse, se basent sur le fait que la fonction de distribution est constante le long des caractéristiques. Il s'agit alors de calculer les caractéristiques puis d'interpoler au pied de celles-ci. Une description plus détaillée est donnée en introduction de la première partie. Les méthodes semi-Lagrangiennes constituent donc un compromis entre les méthodes PIC (où il n'y a pas de projection dans l'espace des phases) et les méthodes purement Eulériennes (limitées à une condition CFL). Ces méthodes sont, entre autres, développées dans les codes GYSELA (GYrokinetic-SEmi LAgrangien) [80, 90] et SELALIB (SEmi-LAgrangian LIBrary) [67].

1.5 Défis

La résolution numérique des équations de Vlasov-Maxwell (ou Vlasov-Poisson) fait intervenir toute une série de défis.

- *Dimension.* Le système de Vlasov-Maxwell est un problème de dimension 7 comprenant le temps ainsi que 6 degrés de liberté pour l'espace des phases ($\mathbf{x} \in \mathbb{R}^3$ et $\mathbf{v} \in \mathbb{R}^3$). Obtenir des simulations numériques fines en un temps de calcul raisonnable est un défi majeur en dimension élevé.
- *Multi-échelle.* Les longueurs caractéristiques des phénomènes physiques peuvent avoir des échelles différentes que ce soit en espace, en vitesse ou en temps. Cela nécessite d'une part un grand nombre de points (en utilisant une grille uniforme) en $\mathbf{x}, \mathbf{v}$ et d'autre part une atteinte des temps longs typiques en physique des plasmas.
- *Conservation.* Les quantités physiques conservées au cours du temps au niveau continu (voir Proposition 1.3.2) doivent idéalement également être conservées au niveau des simulations numériques.

Afin de réaliser de "vraies" simulations du type Drift-Kinetic ou ondes KEEN, plusieurs ingrédients sont nécessaires :

- Mettre en place et analyser (dans un cadre simple) des méthodes d'ordre élevé pour capturer les petites échelles
- Utiliser des supercalculateurs ou des cartes graphiques GPU, ce qui nécessite d'adapter les méthodes existantes

— Utiliser des maillages "adaptatifs" lorsque la zone multi-échelle est bien ciblée

Dans cette thèse, nous aborderons ces différents points. La première partie traitera de l'advection linéaire qui constitue la brique de base pour l'équation de Vlasov-Poisson splittée. Dans la seconde partie, nous présenterons des optimisations en temps de calcul par l'utilisation de cartes graphiques GPU et de maillages non uniformes pour la résolution de l'équation de Vlasov-Poisson. La troisième et dernière partie concernera le modèle gyrocinétique en revisitant l'approximation numérique d'opérateurs spécifiques à ce modèle : les opérateurs de gyromoyenne.

Première partie

Méthodes d'advection

Les méthodes semi-Lagrangiennes, qui ont initialement été développées dans le domaine climatologique, ont été appliquées dans le cadre de l'équation de Vlasov au milieu des années 70 par Cheng et Knorr [1]. Elles ont ensuite été réactualisées par Eric Sonnendrücker à la fin des années 90 [6]. Voir également Shoucri [49] pour un historique de ces méthodes.

Ces méthodes se basent sur le fait que la fonction de distribution est constante le long des caractéristiques. Considérons l'équation de transport

$$\partial_t f + \mathbf{a}(\mathbf{x}, t) \cdot \nabla f = 0$$

où $\mathbf{a}$ est le champ d'advection. Les courbes caractéristiques vérifient l'équation

$$\frac{d\mathbf{X}(t)}{dt} = \mathbf{a}(\mathbf{X}(t), t)$$

et notons $\mathbf{X}(t; \mathbf{x}, s)$ l'unique solution de cette équation différentielle vérifiant $\mathbf{X}(s) = \mathbf{x}$. Le principe est alors de parcourir les caractéristiques en arrière afin de se ramener au temps précédent. Plus précisément,

1. Au temps t_{n+1}, nous calculons le pied de la caractéristique $\mathbf{X}(t_n; \mathbf{x}_i, t_{n+1})$ pour tout point $\mathbf{x}_i$ du maillage.
2. La valeur cherchée $f^{n+1}(\mathbf{x}_i) = f^n(\mathbf{X}(t_n; \mathbf{x}_i, t_{n+1}))$ se calcule par interpolation de la fonction f^n au point $\mathbf{X}(t_n; \mathbf{x}_i, t_{n+1})$. Différentes méthodes d'interpolation peuvent être utilisées.

La figure 1 présente le principe d'une méthode semi-Lagrangienne en dimension 1 et 2.

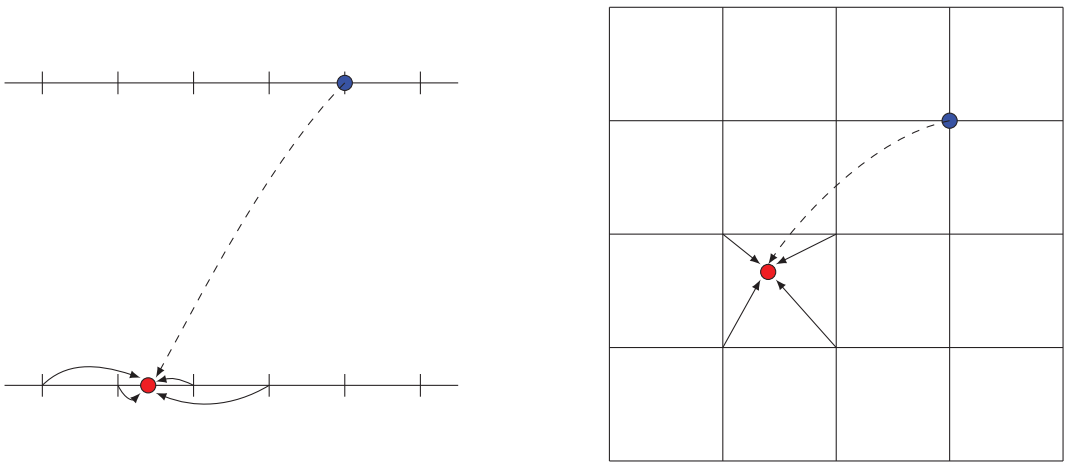


FIGURE 1.2 – Schémas d'une méthode semi-Lagrangienne en dimension 1 et 2.

La méthode semi-Lagrangienne n'est pas assez précise lorsque l'interpolation est de bas degré (typiquement une interpolation linéaire). Une interpolation par splines cubiques ou une reconstruction d'Hermine de degré 3 est souvent utilisée et est un bon compromis entre l'efficacité d'une interpolation de degré relativement faible et les résultats numériques obtenus.

La technique du splitting d'opérateur permet de découpler l'équation de Vlasov-Poisson

$$\frac{\partial f}{\partial t} + \mathbf{v} \cdot \nabla_x f + \frac{q}{m} \mathbf{E} \cdot \nabla_v f = 0,$$

en deux équations d'advection à coefficients constants

$$\frac{\partial f}{\partial t} + \mathbf{v} \cdot \nabla_x f = 0$$

avec $\mathbf{v}$ fixé et

$$\frac{\partial f}{\partial t} + \frac{q}{m} \mathbf{E} \cdot \nabla_v f = 0$$

avec $\mathbf{x}$ fixé. Il est à noter que $\mathbf{E}$ ne varie pas lors de la $2^{ème}$ étape car $\frac{dn}{dt} = 0$ et donc $\frac{d\mathbf{E}}{dt} = 0$ par l'équation de Poisson. Le splitting de Strang (détaillé dans le chapitre 5) est la technique de splitting la plus répandue dû à son bon compromis entre simplicité et efficacité. C'est un schéma d'ordre 2 en temps. D'autres splittings d'ordres plus élevés ont été développés [2]. Ces splittings peuvent être dérivés des méthodes de splitting standards pour les ODE appliqués à l'équation du mouvement des trajectoires dans l'espace des phases [3].

Dans cette première partie, nous nous intéresserons aux méthodes d'advection utilisées dans ces splittings. En particulier, nous établirons les *équations équivalentes* (nommées également *équations aux différences* dans la littérature) qui nous permettent d'étudier et de comparer les propriétés dispersives et dissipatives de ces schémas (chapitre 2). Ensuite, nous étudierons davantage en détail le schéma Galerkin Discontinu Semi-Lagrangien (SLDG) en montrant qu'il possède une propriété de superconvengence en temps long (chapitre 3).

Chapitre 2

Equations équivalentes

Pour les EDP linéaires résolus par des maillages à pas constants, l'analyse de Fourier donne la solution exacte du schéma. Cette méthode n'est plus applicable pour les équations non linéaires ou les équations d'advection à vitesse variable. La méthode de l'équation équivalente qui consiste à déterminer l'EDP équivalente à l'équation aux différences a été introduite (voir [16]) pour d'une part traiter les cas non linéaires et d'autre part fournir des solutions explicites (et non plus sous représentation intégrale) pour les EDP linéaires. L'étude de ces équations équivalentes permet en particulier d'accéder aux propriétés dispersives et dissipatives des schémas et de les comparer entre elles [8].
Nous étudierons en premier lieu des schémas résolvant l'équation d'advection constante. L'étude de l'équation d'advection à vitesse constante est intéressante en soi et lorsque l'équation de Vlasov est splittée, on se ramène à de l'advection constante (voir [87]). Dans la dernière partie, nous étudions un schéma de type Lagrange+Projection utilisé classiquement pour résoudre les équations de l'hydrodynamique.
Les travaux de ce chapitre ont été réalisés en collaboration avec Daniel Bouche et Michel Mehrenberger et la dernière partie a fait l'objet d'un contrat CEA.

2.1 Notion d'équation équivalente

Le cadre d'étude est l'équation d'advection linéaire

$$\begin{cases} \partial_t u(x,t) + a\partial_x u(x,t) = 0, & x \in [0,1],\ t \geq 0 \\ u(x,0) = u_0(x) \end{cases} \tag{2.1.1}$$

avec des conditions aux bords périodiques et une vitesse $a \in \mathbb{R}$ constante. La solution de cette équation est connue de manière explicite : $u(x,t) = u_0(x-at)$. On suppose que cette équation est discrétisée par un schéma à un pas de temps de la forme :

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} + \frac{a}{\Delta x} \sum_{m=-M}^{M} a_m u_{i+m}^n = 0 \tag{2.1.2}$$

où Δx est le pas d'espace, Δt est le pas de temps, $u_i^n \approx u(i\Delta x, n\Delta t)$, $M \in \mathbb{N}^\star$ et les a_m sont des réels. Dans la suite, nous noterons η le nombre de Courant : $\eta = \frac{a\Delta t}{\Delta x}$ et $t_n = n\Delta t$ sera le temps atteint au bout de n itérations du schéma. Commençons par la définition d'une équation équivalente.

Définition 2.1.1. *On appelle équation équivalente d'un schéma l'équation obtenue en ajoutant récursivement au modèle étudié le terme d'ordre dominant de l'erreur de troncature du schéma.*

L'idée est que le schéma va être consistant à un certain ordre p avec l'équation d'advection mais sera consistant à l'ordre $p+r$ avec l'équation équivalente où cette dernière sera composée des termes de l'équation d'advection ainsi que de r termes de "perturbation". L'équation équivalente associée à un schéma est alors de la forme

$$\partial_t u + a\partial_x u + \sum_{i=2}^{r+1} q_i \partial_x^i u = 0.$$

En pratique, on ne considère que les premiers coefficients non nuls, le choix du nombre de termes est peu détaillé dans la littérature. Pour les schémas étudiés ici, on se restreint ainsi aux coefficients q_2, q_3, q_4 et q_5.

Proposition 2.1.2. *Soit le schéma à un pas de temps (2.1.2) résolvant l'équation d'advection linéaire (2.1.1). On pose pour tout entier naturel j,*

$$S_j = \sum_{m=-M}^{M} a_m m^j$$

et on suppose que le schéma est tel que $S_0 = 0$ et $S_1 = 1$. Alors l'équation équivalente associé à ce schéma vaut :

$$\partial_t u + a\partial_x u + a\sum_{j=2}^{5} c_j \frac{(\Delta x)^{j-1}}{j} \partial_x^j u = 0$$

où

$$\begin{aligned} c_2 &= S_2 + \eta \\ c_3 &= (S_3 + 3\eta S_2 + 2\eta^2)/2 \\ c_4 &= (S_4 + \eta(4S_3 + 3S_2^2) + 12\eta^2 S_2 + 6\eta^3)/6 \\ c_5 &= (S_5 + \eta(5S_4 + 10S_2S_3) + \eta^2(20S_3 + 30S_2^2) + 60\eta^3 S_2 + 24\eta^4)/24. \end{aligned}$$

Démonstration. Soit u une solution de

$$\partial_t u + a\partial_x u = 0$$

et $\tilde{u}$ une solution de

$$\partial_t \tilde{u} + a\partial_x \tilde{u} + q_2\partial_x^2 \tilde{u} + q_3\partial_x^3 \tilde{u} + q_4\partial_x^4 \tilde{u} + q_5\partial_x^5 \tilde{u} = 0. \tag{2.1.3}$$

Afin d'étudier l'erreur commise sur un pas de temps, on suppose qu'à l'instant $t_n = n\Delta t$:

$$u(\cdot, t_n) = \tilde{u}(\cdot, t_n).$$

En intégrant (2.1.3) entre t_n et t_{n+1}, on obtient :

$$\tilde{u}(x,t_{n+1}) - \tilde{u}(x,t_n) + a\int_{t_n}^{t_{n+1}} \partial_x\tilde{u}(x,t)dt + q_2\int_{t_n}^{t_{n+1}} \partial_x^2\tilde{u}(x,t)dt +$$
$$q_3\int_{t_n}^{t_{n+1}} \partial_x^3\tilde{u}(x,t)dt + q_4\int_{t_n}^{t_{n+1}} \partial_x^4\tilde{u}(x,t)dt + q_5\int_{t_n}^{t_{n+1}} \partial_x^5\tilde{u}(x,t)dt = 0 \quad (*)$$

Or, par développement de Taylor,

$$\tilde{u}(x,t) = \sum_{k=0}^{5} \frac{(t-t_n)^k}{k!} \partial_t^k \tilde{u}(x,t_n) + \mathcal{O}(\Delta t^6)$$

et en remplaçant les dérivées temporelles $\partial_t \tilde{u}(x,t_n)$ par

$$-a\partial_x \tilde{u}(x,t_n) - q_2 \partial_x^2 \tilde{u}(x,t_n) - q_3 \partial_x^3 \tilde{u}(x,t_n) - q_4 \partial_x^4 \tilde{u}(x,t_n) - q_5 \partial_x^5 \tilde{u}(x,t_n)$$

nous obtenons

$$\begin{aligned}
\tilde{u}(x,t) = {} & \tilde{u}(x,t_n) + (t-t_n)\left[-a\partial_x \tilde{u}(x,t_n) - q_2 \partial_x^2 \tilde{u}(x,t_n) - q_3 \partial_x^3 \tilde{u}(x,t_n) - q_4 \partial_x^4 \tilde{u}(x,t_n)\right] + \\
& \frac{(t-t_n)^2}{2}\left[a^2 \partial_x^2 \tilde{u}(x,t_n) + 2aq_2 \partial_x^3 \tilde{u}(x,t_n) + (2aq_3 + q_2^2)\partial_x^4 \tilde{u}(x,t_n)\right] + \\
& \frac{(t-t_n)^3}{6}\left[-a^3 \partial_x^3 \tilde{u}(x,t_n) - 3a^2 q_2 \partial_x^4 \tilde{u}(x,t_n)\right] + \\
& \frac{(t-t_n)^4}{24}\left[a^4 \partial_x^4 \tilde{u}(x,t_n)\right] + \mathcal{O}(\Delta t^5).
\end{aligned}$$

En revenant à l'égalité $(*)$, nous pouvons maintenant calculer les termes :

$$\begin{aligned}
\int_{t_n}^{t_{n+1}} \partial_x \tilde{u}(x,t)dt = {} & \Delta t \partial_x \tilde{u}(x,t_n) - a\frac{(\Delta t)^2}{2}\partial_x^2 \tilde{u}(x,t_n) + \left(a^2\frac{(\Delta t)^3}{6} - q_2\frac{(\Delta t)^2}{2}\right)\partial_x^3 \tilde{u}(x,t_n) + \\
& \left(-a^3\frac{(\Delta t)^4}{24} + 2aq_2\frac{(\Delta t)^3}{6} - q_3\frac{(\Delta t)^2}{2}\right)\partial_x^4 \tilde{u}(x,t_n) + \\
& \left(-q_4\frac{(\Delta t)^2}{2} + (2aq_3+q_2^2)\frac{(\Delta t)^3}{6} - 3a^2q_2\frac{(\Delta t)^4}{24} + a^4\frac{(\Delta t)^5}{120}\right)\partial_x^5 \tilde{u}(x,t_n) + \mathcal{O}(\Delta t^6) \\
\int_{t_n}^{t_{n+1}} \partial_x^2 \tilde{u}(x,t)dt = {} & \Delta t \partial_x^2 \tilde{u}(x,t_n) - a\frac{(\Delta t)^2}{2}\partial_x^3 \tilde{u}(x,t_n) + \left(a^2\frac{(\Delta t)^3}{6} - q_2\frac{(\Delta t)^2}{2}\right)\partial_x^4 \tilde{u}(x,t_n) + \\
& \left(-a^3\frac{(\Delta t)^4}{24} + 2aq_2\frac{(\Delta t)^3}{6} - q_3\frac{(\Delta t)^2}{2}\right)\partial_x^5 \tilde{u}(x,t_n) + \mathcal{O}(\Delta t^5) \\
\int_{t_n}^{t_{n+1}} \partial_x^3 \tilde{u}(x,t)dt = {} & \Delta t \partial_x^3 \tilde{u}(x,t_n) - a\frac{(\Delta t)^2}{2}\partial_x^4 \tilde{u}(t_n,x) + \\
& \left(a^2\frac{(\Delta t)^3}{6} - q_2\frac{(\Delta t)^2}{2}\right)\partial_x^5 \tilde{u}(x,t_n) + \mathcal{O}(\Delta t^4) \\
\int_{t_n}^{t_{n+1}} \partial_x^4 \tilde{u}(x,t)dt = {} & \Delta t \partial_x^4 \tilde{u}(x,t_n) - a\frac{(\Delta t)^2}{2}\partial_x^5 \tilde{u}(x,t_n) + \mathcal{O}(\Delta t^3) \\
\int_{t_n}^{t_{n+1}} \partial_x^5 \tilde{u}(x,t)dt = {} & \Delta t \partial_x^5 \tilde{u}(x,t_n) + \mathcal{O}(\Delta t^2).
\end{aligned}$$

On remarque que le coefficient devant $\partial_x^j u$ est en $\mathcal{O}((\Delta x)^{j-1})$, ce qui justifie les

développements suivant l'ordre des dérivées. Ainsi, on en déduit

$$\begin{aligned}\tilde{u}(x,t_{n+1}) &= \tilde{u}(x,t_n) - a\Delta t\partial_x\tilde{u}(x,t_n) + \left(a^2\frac{(\Delta t)^2}{2} - q_2\Delta t\right)\partial_x^2\tilde{u}(x,t_n) + \\ &\left(-a^3\frac{(\Delta t)^3}{6} + aq_2(\Delta t)^2 - q_3\Delta t\right)\partial_x^3\tilde{u}(x,t_n) + \\ &\left(a^4\frac{(\Delta t)^4}{24} - 3a^2q_2\frac{(\Delta t)^3}{6} + (2aq_3+q_2^2)\frac{(\Delta t)^2}{2} - q_4\Delta t\right)\partial_x^4\tilde{u}(x,t_n) + \\ &\left(-a^5\frac{(\Delta t)^5}{120} + 4a^3q_2\frac{(\Delta t)^4}{24} + (-3a^2q_3 - 3aq_2^2)\frac{(\Delta t)^3}{6} + \right. \\ &\left.(2q_2q_3 + 2aq_4)\frac{(\Delta t)^2}{2} - q_5\Delta t\right)\partial_x^5\tilde{u}(x,t_n) + \mathcal{O}(\Delta t^6).\end{aligned}$$

Par le même raisonnement avec u et l'équation d'advection $\partial_t u + a\partial_x u = 0$, nous trouvons

$$\begin{aligned}u(x,t_{n+1}) &= u(x,t_n) - a\Delta t\partial_x u(x,t_n) + a^2\frac{(\Delta t)^2}{2}\partial_x^2 u(x,t_n) - a^3\frac{(\Delta t)^3}{6}\partial_x^3 u(x,t_n) + \\ &a^4\frac{(\Delta t)^4}{24}\partial_x^4 u(x,t_n) - a^5\frac{(\Delta t)^5}{120}\partial_x^5 u(x,t_n) + \mathcal{O}(\Delta t^6).\end{aligned}$$

Puisque $\tilde{u}(x,t_n) = u(x,t_n)$, l'erreur de troncature vaut

$$\begin{aligned}\frac{1}{\Delta t}\left(u(x,t_{n+1}) - \tilde{u}(x,t_{n+1})\right) &= q_2\partial_x^2 u(x,t_n) + (q_3 - aq_2\Delta t)\partial_x^3 u(x,t_n) + \\ &\left(q_4 - (2aq_3+q_2^2)\frac{\Delta t}{2} + 3a^2q_2\frac{(\Delta t)^2}{6}\right)\partial_x^4 u(x,t_n) + \\ &\left(q_5 - (q_2q_3+aq_4)\Delta t + (a^2q_3+a_2^2)\frac{(\Delta t)^2}{2}\right. \\ &\left.-a^3q_2\frac{(\Delta t)^3}{6}\right)\partial_x^5 u(x,t_n) + \mathcal{O}(\Delta t)^6.\end{aligned}$$

L'erreur de troncature en fonction du schéma est donnée par

$$\frac{1}{\Delta t}(u(x,t_{n+1}) - u(x,t_n)) + \frac{a}{\Delta x}\sum_{m=-M}^{M} a_m u(x+m\Delta x,t_n)$$

dont le développement vaut

$$\begin{aligned}&\frac{1}{\Delta t}\left(-a\Delta t\partial_x u(x,t_n) + \frac{(a\Delta t)^2}{2}\partial_x^2 u(x,t_n) - \frac{(a\Delta t)^3}{6}\partial_x^3 u(x,t_n)\right. \\ &\left.+\frac{(a\Delta t)^4}{24}\partial_x^4 u(x,t_n) - \frac{(a\Delta t)^5}{120}\partial_x^5 u(x,t_n) + \mathcal{O}(\Delta t^6)\right) + \\ &\frac{a}{\Delta x}\sum_{m=-M}^{M} a_m\left(u(x,t_n) + m\Delta x\partial_x u(x,t_n) + \frac{(m\Delta x)^2}{2}\partial_x^2 u(x,t_n) + \right. \\ &\left.\frac{(m\Delta x)^3}{6}\partial_x^3 u(x,t_n) + \frac{(m\Delta x)^4}{24}\partial_x^4 u(x,t_n) + \frac{(m\Delta x)^5}{120}\partial_x^5 u(x,t_n) + \mathcal{O}(\Delta x^6)\right)\end{aligned}$$

En identifiant les coefficients des développements des deux erreurs précédentes, on en déduit l'expression des coefficients de l'équation équivalente :

$$\begin{aligned} q_2 &= \frac{a\Delta x[\eta + S_2]}{2} \\ q_3 &= \frac{a(\Delta x)^2[S_3 + 3\eta S_2 + 2\eta^2]}{6} \\ q_4 &= \frac{a(\Delta x)^3[S_4 + \eta(4S_3 + 3S_2^2) + 12\eta^2 S_2 + 6\eta^3]}{24} \\ q_5 &= \frac{a(\Delta x)^4[S_5 + \eta(5S_4 + 10S_2S_3) + \eta^2(20S_3 + 30S_2^2) + 60\eta^3 S_2 + 24\eta^4]}{120} \end{aligned}$$

□

Ces formules donnent donc directement les coefficients de l'équation équivalente à partir des coefficients du schéma numérique. Les conditions $S_0 = 0$ et $S_1 = 1$ dans l'énoncé de la proposition sont naturelles : elles signifient que la méthode est consistante avec l'équation à résoudre d'une part et que l'erreur de consistance ne comporte pas de terme en Δx. Lorsque, dans le premier terme de l'équation équivalente $\partial_x^i u$, i est impair, le schéma sera davantage dispersif au sens des ondes ; il y a alors une erreur dans la vitesse de propagation des ondes ce qui crée des oscillations aux points de discontinuités, par exemple pour une marche d'escalier. Dans le cas contraire, le schéma sera davantage dissipatif (ou diffusif) et aura tendance à lisser les profils (voir [8]).

2.2 Applications aux schémas résolvant l'équation d'advection

Nous allons utiliser les résultats de la partie précédente afin de comparer différents schémas résolvant l'équation d'advection.

2.2.1 Schémas de Lax-Wendroff et Warming-Beam

Nos premiers exemples sont les schémas classiques de Lax-Wendroff [15] et Warming-Beam [7]. Rappelons le principe de ces schémas. Nous supposons que $0 < \eta < 1$ et que les $(u_i^n)_{i\in\{0,\dots,N\}}$ à l'instant t_n soient connus. Pour calculer u_i^{n+1}, nous construisons le polynôme d'interpolation de Lagrange φ_i^n au temps t_n et aux points x_{i-1}, x_i, x_{i+1} dans le cas du schéma de Lax-Wendroff et aux points x_{i-2}, x_{i-1}, x_i dans le cas du schéma de Warming-Beam. On évalue finalement :

$$u_i^{n+1} = \varphi_i^n(x_i - a\Delta t).$$

Ceci conduit aux formulations suivantes :

$$\begin{aligned} &\text{[Lax-Wendroff] :} && \tfrac{u_i^{n+1}-u_i^n}{\Delta t} + \tfrac{a}{2\Delta x}(u_{i+1}^n - {}_{i-1}^n) - \tfrac{a\eta}{2\Delta x}(u_{i+1}^n - 2u_i^n + u_{i-1}^n) = 0 \\ &\text{[Warming-Beam] :} && \tfrac{u_i^{n+1}-u_i^n}{\Delta t} + \tfrac{a}{2\Delta x}(1-\eta)u_{i-2}^n + \tfrac{a}{\Delta x}(\eta-2)u_{i-1}^n - \tfrac{a}{2\Delta x}(\eta-3)u_i^n = 0. \end{aligned}$$

Ainsi, d'après la Proposition 2.1.2, les équations équivalentes associées à ces schémas valent :

$$\text{[Lax-Wendroff]} : \quad \frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} + \frac{a(\Delta x)^2}{6}(1-\eta^2)\frac{\partial^3 u}{\partial x^3} + \frac{a(\Delta x)^3}{8}\eta(1-\eta^2)\frac{\partial^4 u}{\partial x^4} = 0.$$

$$\text{[Warming-Beam]} : \quad \frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} + \frac{a(\Delta x)^2}{6}(-\eta^2+3\eta-2)\frac{\partial^3 u}{\partial x^3} + \frac{a(\Delta x)^3}{8}(-\eta^3+4\eta^2-5\eta+2)\frac{\partial^4 u}{\partial x^4} = 0.$$

La validation numérique des équations équivalentes fait l'objet de la Figure 2.1. On compare d'une part la solution du schéma (courbe verte) et la solution de l'équation équivalente discrétisée par différences finies (courbe rouge). On observe que ces courbes sont très proches ce qui permet de valider numériquement les équations équivalentes obtenues. La solution exacte de l'équation d'advection (ici en bleue) correspond à la condition initiale puisque nous faisons un nombre entier de tours du domaine spatial. La discrétisation par différences finies de l'équation équivalente est réalisée avec un pas d'espace $\Delta x'$ tel que $\Delta x' << \Delta x$ afin de ne pas perturber les termes de l'équation équivalente. Dans les Figures 2.2 et 2.3, nous avons gardé uniquement le terme en $\frac{\partial^3 u}{\partial x^3}$ (resp. $\frac{\partial^4 u}{\partial x^4}$) des équations équivalentes des schémas de Lax-Wendroff et Warmimg-Beam. On observe alors l'apport du principal terme dispersif (resp. dissipatif).

2.2.2 Schémas semi-Lagrangiens conservatifs

Des méthodes conservatives pour la résolution numérique des équations de Vlasov-Poisson sont développées dans [87] dans un contexte de splitting unidimensionnel. Dans cette partie, nous calculons les équations équivalentes de ces schémas dans le cas de l'advection constante. Nous commençons par reprendre leur description. Les inconnues de ces schémas sont les valeurs moyennes de la fonction de distribution sur une cellule :

$$\bar{u}_i^{n+1} = \frac{1}{\Delta x}\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} u(y, t_{n+1})dy.$$

Par conservation du volume, on a (voir Figure 2.4) :

$$\bar{u}_i^{n+1} = \frac{1}{\Delta x}\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} u(y, t_{n+1})dy = \frac{1}{\Delta x}\int_{x_{i^*-\frac{1}{2}}+\alpha\Delta x}^{x_{i^*+\frac{1}{2}}+\alpha\Delta x} u(y, t_n)dy$$

où i^* est l'entier tel que $x_{i-\frac{1}{2}} - a\Delta t \in \left[x_{i^*-\frac{1}{2}}, x_{i^*+\frac{1}{2}}\right[$ et $\alpha \in [0,1[$ le réel défini par :

$$\alpha\Delta x = (x_{i-\frac{1}{2}} - a\Delta t) - x_{i^*-\frac{1}{2}}.$$

Soit U_i^n la fonction définie par

$$U_i^n(x) = \frac{1}{\Delta x}\int_{x_{i-\frac{1}{2}}}^{x_{i-\frac{1}{2}}+x\Delta x} u(y, t_n)dy.$$

Nous obtenons alors

$$\bar{u}_i^{n+1} = \frac{1}{\Delta x}\int_{x_{i^*-\frac{1}{2}}+\alpha\Delta x}^{x_{i^*+\frac{1}{2}}+\alpha\Delta x} u(y, t_n)dy = \bar{u}_{i^*}^n + U_{i^*+1}^n(\alpha) - U_{i^*}^n(\alpha).$$

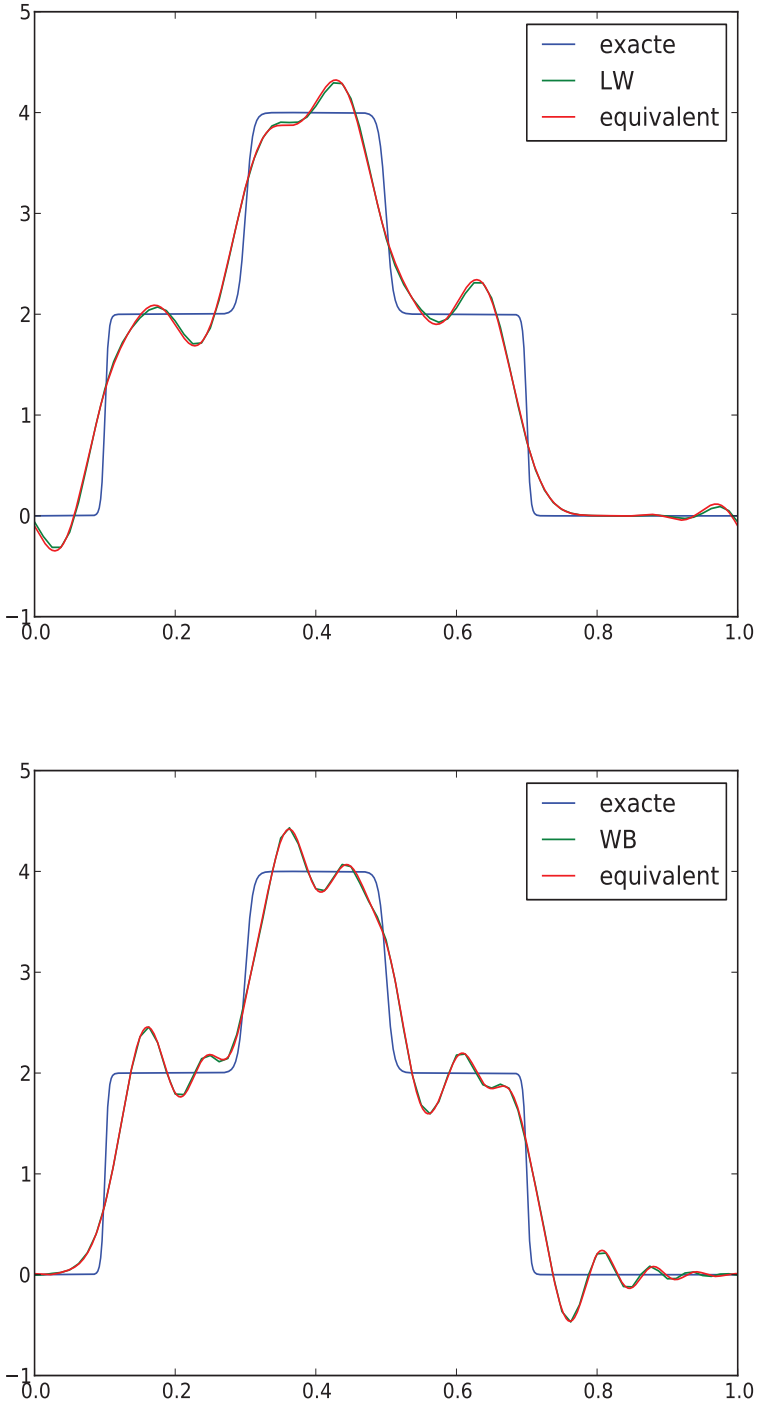


FIGURE 2.1 – Profils de la solution exacte (en bleu), de la solution du schéma (en vert) de Lax-Wendroff (en haut) et Warming-Beam (en bas) et la solution discrétisée par différences finies de l'équation équivalente associée (en rouge). $\Delta x = 1/80$, $\Delta t = 0.01$, $a = 1$.

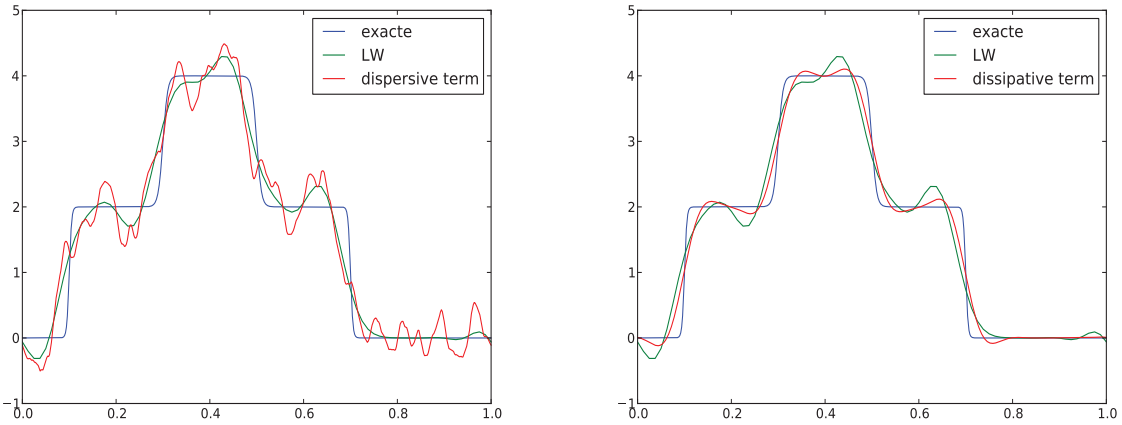


FIGURE 2.2 – Profils de la solution exacte (en bleu), de la solution du schéma de Lax-Wendroff (en vert) et la solution discrétisée par différences finies de l'équation équivalente (en rouge) en considérant uniquement le terme en $\partial_x^3 u$ (à gauche) et en considérant uniquement le terme en $\partial_x^4 u$ (à droite). $\Delta x = 1/80$, $\Delta t = 0.01$, $a = 1$.

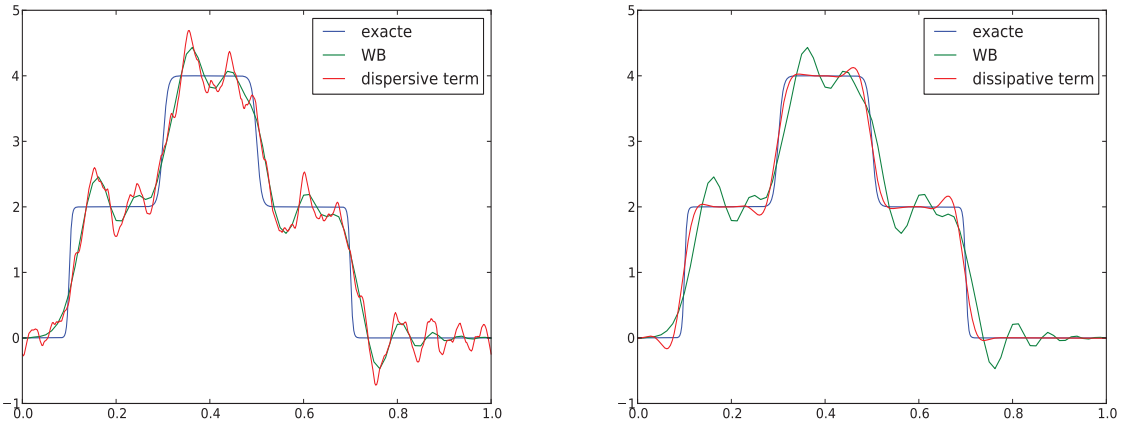


FIGURE 2.3 – Profils de la solution exacte (en bleu), de la solution du schéma de Warmimg-Beam (en vert) et la solution discrétisée par différences finies de l'équation équivalente (en rouge) en considérant uniquement le terme en $\partial_x^3 u$ (à gauche) et en considérant uniquement le terme en $\partial_x^4 u$ (à droite). $\Delta x = 1/80$, $\Delta t = 0.01$, $a = 1$.

Il reste à évaluer $U^n_{i^*+1}(\alpha)$ et $U^n_{i^*}(\alpha)$. Soient $i, n \in \mathbb{N}$, nous avons que

$$U_i^n(0) = 0, \quad U_i^n(1) = \bar{u}_i^n$$

et

$$U_i^{n'}(x) = u(x_{i-\frac{1}{2}} + x\Delta x, t_n)\Delta x, \quad U_i^{n'}(0) = u^n_{i-\frac{1}{2}}\Delta x, \quad U_i^{n'}(1) = u^n_{i+\frac{1}{2}}\Delta x.$$

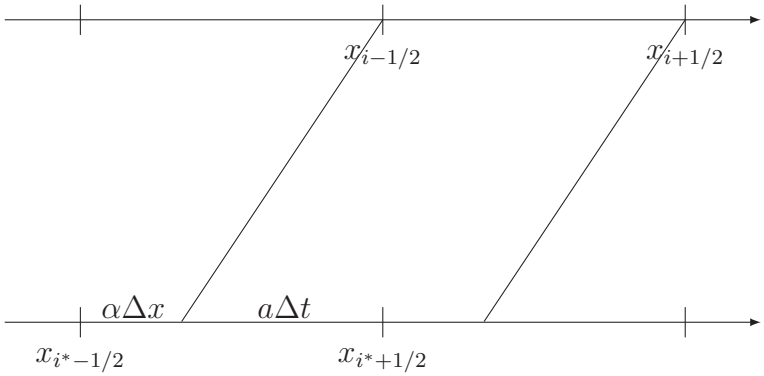


FIGURE 2.4 – Méthode semi-Lagrangienne.

Nous disposons de 4 conditions sur U_i^n, nous procédons alors à une reconstruction polynomiale de U_i^n de degré 3 :

$$U_i^n(x) = x(1-x)^2 u^n_{i-\frac{1}{2}}\Delta x + x^2(x-1)u^n_{i+\frac{1}{2}}\Delta x + x^2(3-2x)\bar{u}_i^n.$$

Rappelons que nous cherchons à calculer les intégrales $\bar{u}_i^{n+1}$ et que les valeurs aux noeuds du maillage $u^n_{i-\frac{1}{2}}$ et $u^n_{i+\frac{1}{2}}$ sont inconnues. Nous allons donc donner plusieurs reconstructions de $u^n_{i-\frac{1}{2}}$ et $u^n_{i+\frac{1}{2}}$ sur la cellule i en fonction des valeurs $(\bar{u}_i^n)_{i\in[0,N-1]}$.

— Reconstruction de Lagrange de degré 3 (LAG3) :

$$u^n_{i-1/2}\Delta x = \frac{5}{6}\bar{u}_i^n - \frac{1}{6}\bar{u}^n_{i+1} + \frac{1}{3}\bar{u}^n_{i-1}, \quad u^n_{i+1/2}\Delta x = \frac{5}{6}\bar{u}_i^n + \frac{1}{3}\bar{u}^n_{i+1} - \frac{1}{6}\bar{u}^n_{i-1}$$

— Reconstruction PPM0 :

$$u^n_{i-1/2}\Delta x = \frac{1}{2}(\bar{u}^n_{i-1} + \bar{u}_i^n), \quad u^n_{i+1/2}\Delta x = \frac{1}{2}(\bar{u}_i^n + \bar{u}^n_{i+1})$$

— Reconstruction PPM1 :

$$u^n_{i-1/2}\Delta x = \frac{7}{12}(\bar{u}^n_{i-1}+\bar{u}_i^n) - \frac{1}{12}(\bar{u}^n_{i-2}+\bar{u}^n_{i+1}), \quad u^n_{i+1/2}\Delta x = \frac{7}{12}(\bar{u}_i^n+\bar{u}^n_{i+1}) - \frac{1}{12}(\bar{u}^n_{i-1}+\bar{u}^n_{i+2})$$

— Reconstruction PPM2 :

$$u^n_{i-1/2}\Delta x = \frac{1}{60}((\bar{u}^n_{i-3} + \bar{u}^n_{i+2}) - 9(\bar{u}^n_{i-2} + \bar{u}^n_{i+1}) + 45(\bar{u}^n_{i-1} + \bar{u}_i^n))$$

$$u^n_{i+1/2}\Delta x = \frac{1}{60}((\bar{u}^n_{i-2} + \bar{u}^n_{i+3}) - 9(\bar{u}^n_{i-1} + \bar{u}^n_{i+2}) + 45(\bar{u}_i^n + \bar{u}^n_{i+1}))$$

— Reconstruction PSM : les valeurs aux noeuds du maillage sont solutions du système presque tridiagonal :

$$u^n_{i-1/2} + 4u^n_{i+1/2} + u^n_{i+3/2} = \frac{3}{\Delta x}(\bar{u}_i^n + \bar{u}^n_{i+1}).$$

Calcul de l'équation équivalente du schéma Lagrange 3.

Nous allons calculer une équation équivalente pour le schéma Lagrange 3 de deux façons différentes : par calcul direct et par combinaison des équations équivalentes des schémas Lax-Wendroff et Warming-Beam.

• Calcul direct

Le schéma de Lagrange 3 est caractérisé par les coefficients (suivant la formulation 2.1.2) :

$$a_{-2} = \frac{1-\eta^2}{6},\ a_{-1} = \frac{\eta^2-\eta-2}{2},\ a_0 = \frac{-\eta^2+2\eta+1}{2},\ a_1 = \frac{(1-\eta)(2-\eta)}{6},\ a_2 = 0.$$

Ainsi, d'après la Proposition 2.1.2, une équation équivalente pour le schéma Lagrange 3 vaut :

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} + \frac{a(\Delta x)^3}{24}(\eta^3-2\eta^2-\eta+2)\frac{\partial^4 u}{\partial x^4} + \frac{a(\Delta x)^4}{60}(2\eta^4-5\eta^3+5\eta-2)\frac{\partial^5 u}{\partial x^5} = 0.$$

• A partir des équations équivalentes de Lax-Wendroff et Warming-Beam

Le schéma Lagrange 3 est obtenu comme combinaison convexe des schémas de Lax-Wendroff et Warming-Beam :

$$LAG3 = \left(1 - \frac{1+\eta}{3}\right) LW + \frac{1+\eta}{3} WB.$$

La combinaison convexe appliquée aux équations équivalentes va annuler le coefficient devant $(\Delta x)^2$:

$$(1-\eta^2)(1-\frac{1+\eta}{3}) + (-\eta^2+3\eta-2)\frac{1+\eta}{3} = 0.$$

Il va donc falloir écrire un terme supplémentaire dans les équations équivalentes pour Lax-Wendroff et Warming-Beam.

Pour Lax-Wendroff, nous obtenons :

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} + \frac{a(\Delta x)^2}{6}(1-\eta^2)\frac{\partial^3 u}{\partial x^3} + \frac{a(\Delta x)^3}{8}\eta(1-\eta^2)\frac{\partial^4 u}{\partial x^4} + \frac{a(\Delta x)^4}{120}(-6\eta^4+5\eta^2+1)\frac{\partial^5 u}{\partial x^5} = 0$$

et pour Warming-Beam :

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} + \frac{a(\Delta x)^2}{6}(-\eta^2+3\eta-2)\frac{\partial^3 u}{\partial x^3} + \frac{a(\Delta x)^3}{8}(-\eta^3+4\eta^2-5\eta+2)\frac{\partial^4 u}{\partial x^4} + \frac{a(\Delta x)^4}{120}(-6\eta^4+30\eta^3-55\eta^2+45\eta-14)\frac{\partial^5 u}{\partial x^5} = 0$$

Nous retrouvons alors l'équation équivalente pour le schéma Lagrange 3 :

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} + \frac{a(\Delta x)^3}{24}(\eta^3-2\eta^2-\eta+2)\frac{\partial^4 u}{\partial x^4} + \frac{a(\Delta x)^4}{60}(2\eta^4-5\eta^3+5\eta-2)\frac{\partial^5 u}{\partial x^5} = 0.$$

La Figure 2.5 valide numériquement cette équation équivalente.

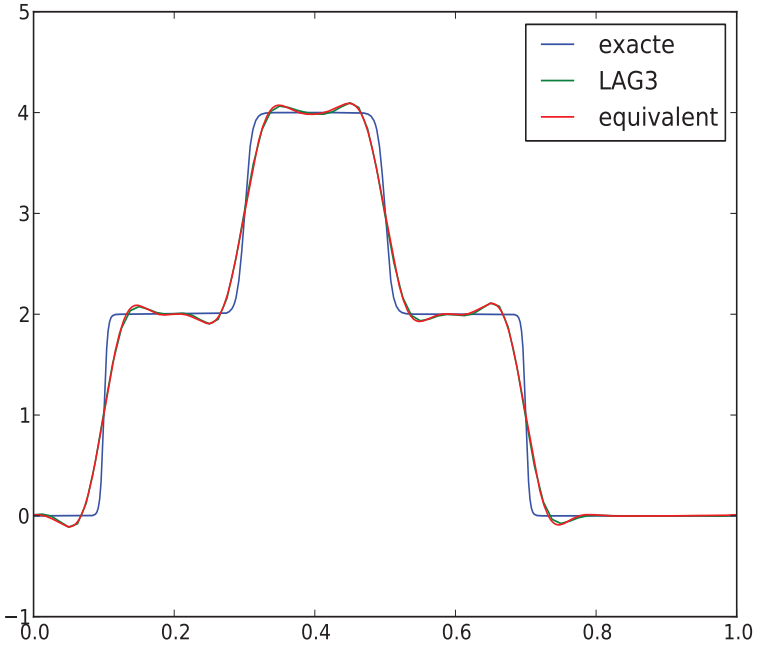


FIGURE 2.5 – Profils de la solution exacte (en bleu), de la solution du schéma LAG3 (en vert) et la solution discrétisée par différences finies de l'équation équivalente associée (en rouge). $\Delta x = 1/80$, $\Delta t = 0.01$, $a = 1$.

Equations équivalentes pour les schémas PPM0, PPM1 et PPM2

Les différents schémas sont caractérisés par les coefficients (suivant la formulation 2.1.2) :

$$
\begin{aligned}
PPM0 : \quad & a_{-2} = \tfrac{-\eta^2+\eta}{2}, \quad a_{-1} = \tfrac{3\eta^2-4\eta-1}{2}, \\
& a_0 = \tfrac{-3\eta^2+5\eta}{2}, \quad a_1 = \tfrac{\eta^2-2\eta+1}{2}, \quad a_2 = 0. \\
PPM1 : \quad & a_{-3} = \tfrac{\eta^2-\eta}{12}, \quad a_{-2} = \tfrac{-7\eta^2+6\eta+1}{12}, \quad a_{-1} = \tfrac{4\eta^2-5\eta-2}{3}, \\
& a_0 = \tfrac{-4\eta^2+7\eta}{3}, \quad a_1 = \tfrac{7\eta^2-15\eta+8}{12}, \quad a_2 = \tfrac{-\eta^2+2\eta-1}{12}, \quad a_3 = 0. \\
PPM2 : \quad & a_{-4} = \tfrac{-\eta^2+\eta}{60}, \quad a_{-3} = \tfrac{8\eta^2-7\eta-1}{60}, \quad a_{-2} = \tfrac{-12\eta^2+9\eta+3}{20}, \\
& a_{-1} = \tfrac{5\eta^2-6\eta-3}{4}, \quad a_0 = \tfrac{-5\eta^2+9\eta}{4}, \quad a_1 = \tfrac{12\eta^2-27\eta+15}{20}, \\
& a_2 = \tfrac{-8\eta^2+17\eta-9}{60}, \quad a_3 = \tfrac{\eta^2-2\eta+1}{60}, \quad a_4 = 0.
\end{aligned}
$$

Ainsi, d'après la Proposition 2.1.2, les équations équivalentes valent :

$$
\begin{aligned}
PPM0 : \quad & \tfrac{\partial u}{\partial t} + a\tfrac{\partial u}{\partial x} + \tfrac{a(\Delta x)^2}{6}(2\eta^2 - 3\eta + 1)\tfrac{\partial^3 u}{\partial x^3} + \tfrac{a(\Delta x)^3}{8}(3\eta^3 - 6\eta^2 + 3\eta)\tfrac{\partial^4 u}{\partial x^4} = 0. \\
PPM1 : \quad & \tfrac{\partial u}{\partial t} + a\tfrac{\partial u}{\partial x} + \tfrac{a(\Delta x)^3}{24}(\eta^3 - 2\eta^2 + \eta)\tfrac{\partial^4 u}{\partial x^4} + \tfrac{a(\Delta x)^4}{60}(2\eta^4 - 5\eta^3 + 5\eta - 2)\tfrac{\partial^5 u}{\partial x^5} = 0. \\
PPM2 : \quad & \tfrac{\partial u}{\partial t} + a\tfrac{\partial u}{\partial x} + \tfrac{a(\Delta x)^3}{24}(\eta^3 - 2\eta^2 + \eta)\tfrac{\partial^4 u}{\partial x^4} + \tfrac{a(\Delta x)^4}{60}(2\eta^4 - 5\eta^3 + 4\eta^2 - \eta)\tfrac{\partial^5 u}{\partial x^5} = 0.
\end{aligned}
$$

La Figure 2.6 valide numériquement l'équation équivalente pour le schéma PPM0. Par contre, la Figure 2.7 montre que deux termes dans l'équation équivalente ne sont pas suffisants pour

rendre compte du schéma. Afin d'obtenir un graphique concluant, il faut considérer le terme suivant et ainsi ajouter le terme d'ordre 6 :

$$\frac{a(\Delta x)^5}{144}(2\eta^5 - 6\eta^4 - 3\eta^3 + 16\eta^2 - 9\eta)\frac{\partial^6 u}{\partial x^6}.$$

De même, dans le cas du schéma PPM2 (Figure 2.8), 5 termes de l'équation équivalente sont nécessaires afin d'obtenir une bonne correspondance.

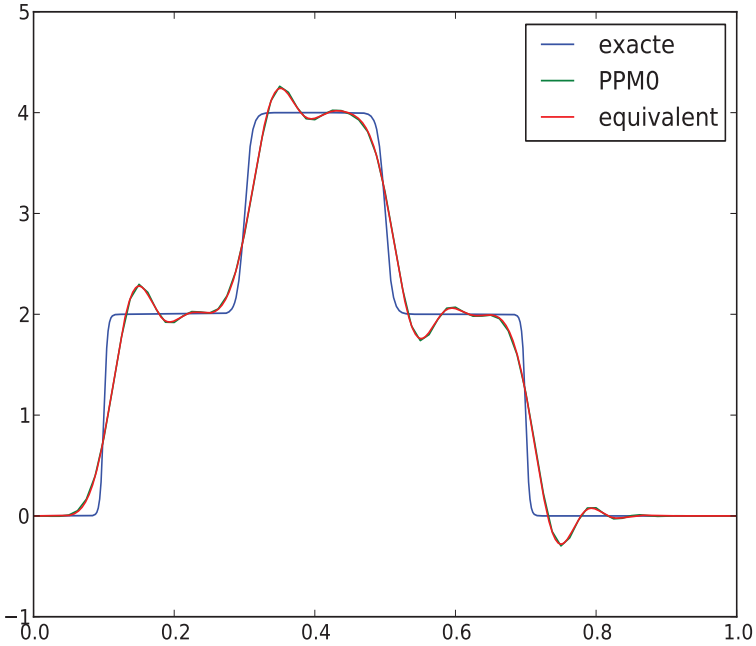


FIGURE 2.6 – Profils de la solution exacte (en bleu), de la solution du schéma PPM0 (en vert) et la solution discrétisée par différences finies de l'équation équivalente associée (en rouge). $\Delta x = 1/80$, $\Delta t = 0.01$, $a = 1$.

Equation équivalente pour le schéma PSM

Rappelons que N désigne le nombre de mailles en espace. Pour le schéma PSM, les valeurs ponctuelles sont solutions du système :

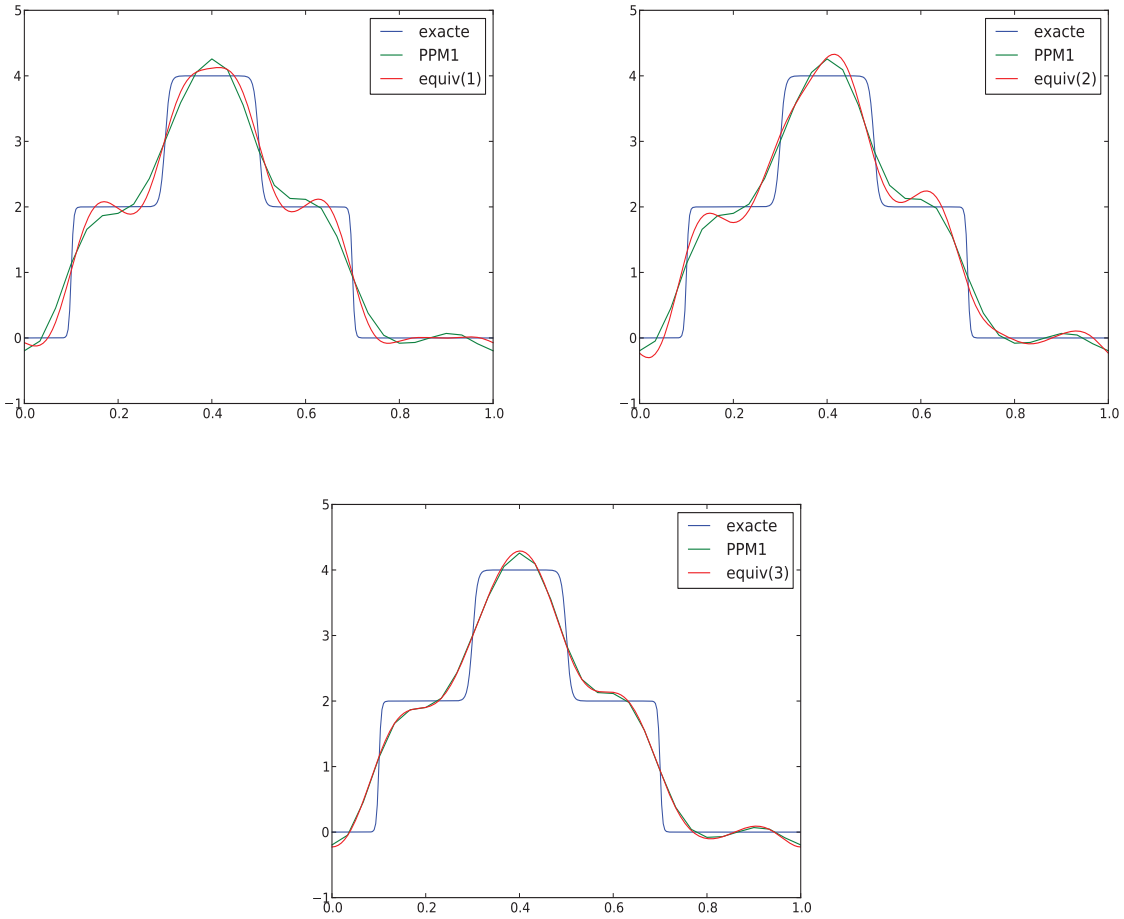


FIGURE 2.7 – Profils de la solution exacte (en bleu), de la solution du schéma PPM1 (en vert) et la solution discrétisée par différences finies de l'équation équivalente (en rouge) avec un terme dans l'équation équivalente (en haut à gauche) deux termes (en haut à droite) et trois termes (en bas). $\Delta x = 1/80$, $\Delta t = 0.01$, $a = 1$.

$$\begin{pmatrix} 4 & 1 & 0 & \dots & 0 & 1 \\ 1 & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & 1 \\ 1 & 0 & \dots & 0 & 1 & 4 \end{pmatrix} \begin{pmatrix} u_{1/2} \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ u_{N-1/2} \end{pmatrix} = 3 \begin{pmatrix} \bar{u}_0^n + \bar{u}_1^n \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \bar{u}_{N-1}^n + \bar{u}_N^n \end{pmatrix}$$

Nous remarquons que la matrice du système est circulante et inversible d'inverse la matrice de terme général (voir [14]) :

$$m_{i,j} = \frac{1}{2(1-\tau_N)} \left(\mu_{N-1-|j-i|} + \mu_{|j-i|-1} \right)$$

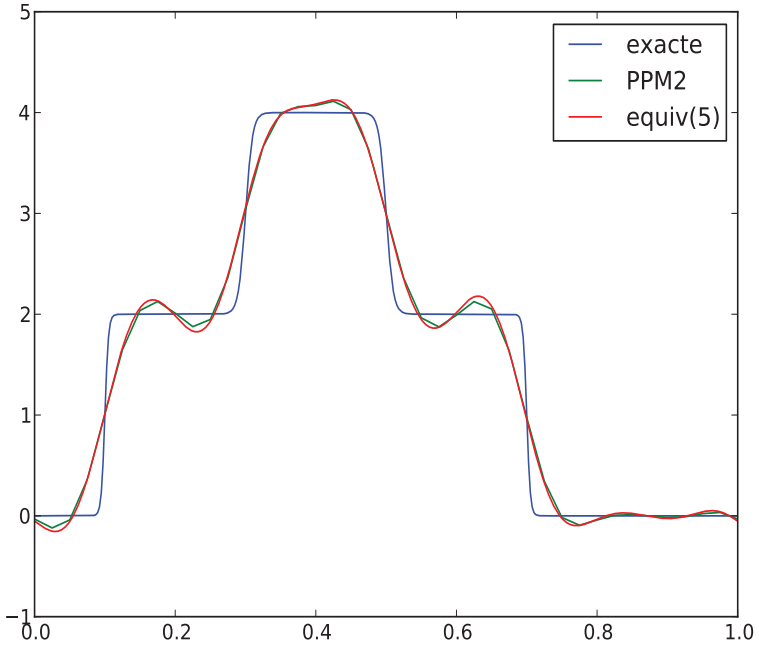


FIGURE 2.8 – Profils de la solution exacte (en bleu), de la solution du schéma PPM2 (en vert) et la solution discrétisée par différences finies de l'équation équivalente avec 5 termes (en rouge). $\Delta x = 1/80$, $\Delta t = 0.01$, $a = 1$.

où τ_k (resp μ_k) désigne la valeur du polynôme de Tchebychev de première (resp. seconde) espèce d'ordre k évalué en -2 (voir l'Appendice E).

Nous en déduisons une expression pour $i = 1, ..., N$:

$$\begin{aligned} u^n_{i-1/2} &= \frac{3}{2(1-\tau_N)} \sum_{j=1}^{N} (\mu_{N-1-|j-i|} + \mu_{|j-i|-1})(\bar{u}^n_{j-1} + \bar{u}^n_j) \\ &= \frac{3}{2(1-\tau_N)} [\bar{u}^n_0(\mu_{N-i} + \mu_{i-2}) + \bar{u}^n_N(\mu_{i-1} + \mu_{N-i-1}) \\ &+ \sum_{j=1}^{N-1} \bar{u}^n_j(\mu_{N-1-|j-i|} + \mu_{|j-i|-1} + \mu_{N-1-|j-i+1|} + \mu_{|j-i+1|-1})]. \end{aligned}$$

Par périodicité $\bar{u}^n_0 = \bar{u}^n_N$, d'où nous avons la formule :

$$u^n_{i-1/2} = \frac{3}{2(1-\tau_N)} \sum_{j=0}^{N-1} \bar{u}^n_j(\mu_{N-1-|j-i|} + \mu_{|j-i|-1} + \mu_{N-1-|j-i+1|} + \mu_{|j-i+1|-1}).$$

Fixons i, n et étudions le coefficient devant $\bar{u}^n_j$ qui vaut

$$\frac{3}{2} \times \frac{\mu_{N-1-|j-i|} + \mu_{|j-i|-1} + \mu_{N-1-|j-i+1|} + \mu_{|j-i+1|-1}}{1-\tau_N}.$$

Nous posons

$$\begin{aligned} num(i,j,N) &= \mu_{N-1-|j-i|} + \mu_{|j-i|-1} + \mu_{N-1-|j-i+1|} + \mu_{|j-i+1|-1} \\ dén(N) &= 1 - \tau_N. \end{aligned}$$

Dans l'Appendice E, nous avons montré que

$$\begin{aligned} \tau_n &= \frac{(-1)^n}{2}\left((2+\sqrt{3})^n + (2-\sqrt{3})^n\right) \\ \mu_n &= \frac{(-1)^n}{2\sqrt{3}}\left((2+\sqrt{3})^{n+1} - (2-\sqrt{3})^{n+1}\right). \end{aligned}$$

Pour un écart $|j-i|$ fixé, on en déduit les équivalents suivants lorsque $N \to +\infty$:

$$\begin{aligned} num(i,j,N) &\sim \frac{(-1)^{N-1-|j-i|}}{2\sqrt{3}}\left((2+\sqrt{3})^{N-|j-i|} - (2+\sqrt{3})^{N-|j-i+1|}\right) \\ dén(N) &\sim \frac{(-1)^{N+1}}{2}(2+\sqrt{3})^N. \end{aligned}$$

Ainsi, le coefficient devant $\bar{u}_j^n$ est équivalent à

$$(-1)^{|j-i|} \times \frac{\sqrt{3}}{2} \times \left((2+\sqrt{3})^{-|j-i|} - (2+\sqrt{3})^{-|j-i+1|}\right).$$

— Pour $j = i$ ou $j = i-1$, ce terme vaut $\frac{3-\sqrt{3}}{2}$.
— Pour $j = i+1$ ou $j = i-2$, ce terme vaut $\frac{5\sqrt{3}-9}{2}$.
— Pour $j = i+2$ ou $j = i-3$, ce terme vaut $\frac{33-19\sqrt{3}}{2}$.

L'approximation faite avec PSM est assez proche de l'approximation faite avec PPM2 :

$$\begin{cases} u_{i-1/2}^n \underset{PPM2}{=} \frac{37}{60}(\bar{u}_{i-1}^n + \bar{u}_i^n) - \frac{8}{60}(\bar{u}_{i-2}^n + \bar{u}_{i+1}^n) + \frac{1}{60}(\bar{u}_{i-3}^n + \bar{u}_{i+2}^n) \\ \\ u_{i-1/2}^n \underset{PSM}{=} \frac{3-\sqrt{3}}{2}(\bar{u}_{i-1}^n + \bar{u}_i^n) + \frac{5\sqrt{3}-9}{2}(\bar{u}_{i-2}^n + \bar{u}_{i+1}^n) + \frac{33-19\sqrt{3}}{2}(\bar{u}_{i-3}^n + \bar{u}_{i+2}^n) \end{cases}$$

puisque

$$\begin{aligned} \frac{37}{60} \approx 0,616 \quad &, \quad \frac{3-\sqrt{3}}{2} \approx 0,634 \\ -\frac{8}{60} \approx -0,133 \quad &, \quad \frac{5\sqrt{3}-9}{2} \approx -0,170 \\ \frac{1}{60} \approx 0,017 \quad &, \quad \frac{33-19\sqrt{3}}{2} \approx 0,046. \end{aligned}$$

Ce phénomène de similitude entre PPM2 et PSM se retrouve largement sur les simulations numériques. L'équation équivalente "limite" pour le schéma PSM en utilisant la formulation ci-dessus vaut :

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} + \frac{a(\Delta x)^3}{24}(\eta^3 - 2\eta^2 + \eta)\frac{\partial^4 u}{\partial x^4} + \frac{a(\Delta x)^4}{180}(6\eta^4 - 15\eta^3 + 10\eta^2 - 1)\frac{\partial^5 u}{\partial x^5} = 0$$

et la vérification numérique est donnée par la Figure 2.9 avec 1 (resp. 2,3) termes de l'équation équivalente.

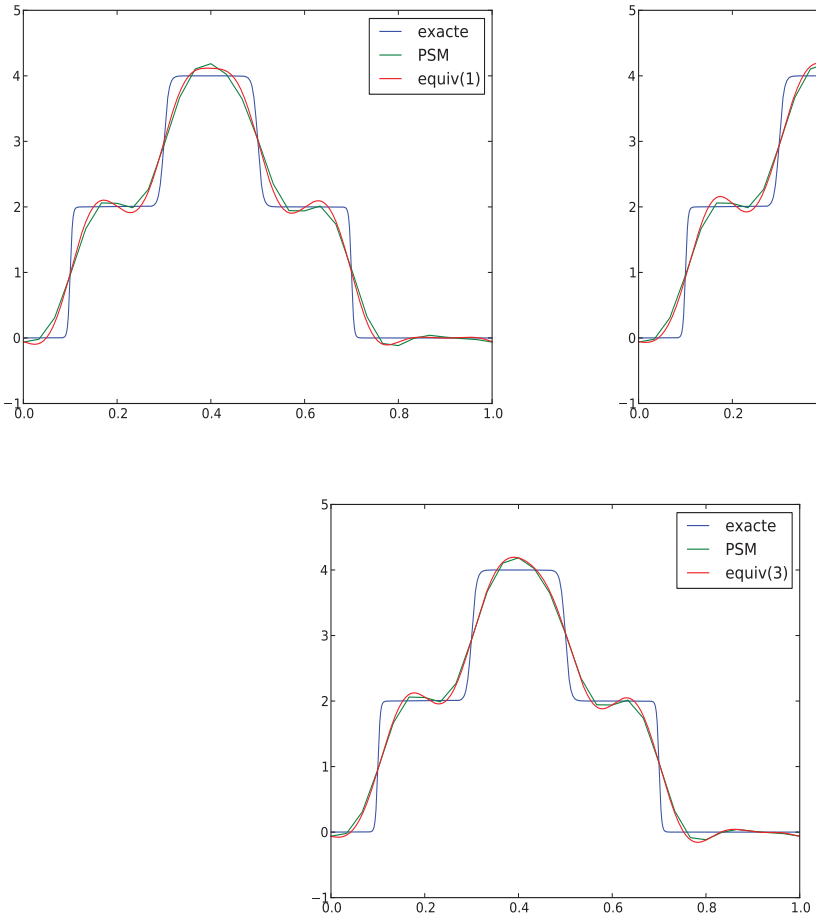


FIGURE 2.9 – Profils de la solution exacte (en bleu), de la solution du schéma PSM (en vert) et la solution discrétisée par différences finies de l'équation équivalente (en rouge) avec un terme dans l'équation équivalente (en haut à gauche) deux termes (en haut à droite) et trois termes (en bas). $\Delta x = 1/80$, $\Delta t = 0.01$, $a = 1$.

2.2.3 Comparaison des différents schémas

Voici un récapitulatif des différents schémas étudiés précédemment avec une équation équivalente sous la forme

$$\partial_t u + a\partial_x u + a(\Delta x)^{n-1}\frac{c_n}{n}\partial_x^n u + a(\Delta x)^n \frac{c_{n+1}}{n+1}\partial_x^{n+1} u = 0.$$

Une telle analyse a déjà été réalisée par exemple dans [8] avec d'autres schémas résolvant l'équation d'advection.

Schémas d'ordre 2

	c_3	c_4
LW	$(1-\eta^2)/2$	$\eta(1-\eta^2)/2$
WB	$(-\eta^2+3\eta-2)/2$	$(-\eta^3+4\eta^2-5\eta+2)/2$
PPM0	$(2\eta^2-3\eta+1)/2$	$(3\eta^3-6\eta^2+3\eta)/2$

La Figure 2.10 présente les valeurs absolues des coefficients de dispersion $|c_3|$ et de dissipation $|c_4|$ en fonction de μ. Nous constatons que les coefficients c_3 et c_4 s'annulent pour $\eta = 1$: les schémas deviennent alors de simples translations résolvant exactement l'équation d'advection. Par ailleurs, les coefficients de dissipation $|c_4|$ tendent vers 0 lorsque η tend vers 0 pour les schémas LW et PPM0. Par contre, le schéma WB peut être fortement dissipatif pour de faibles valeurs de η.

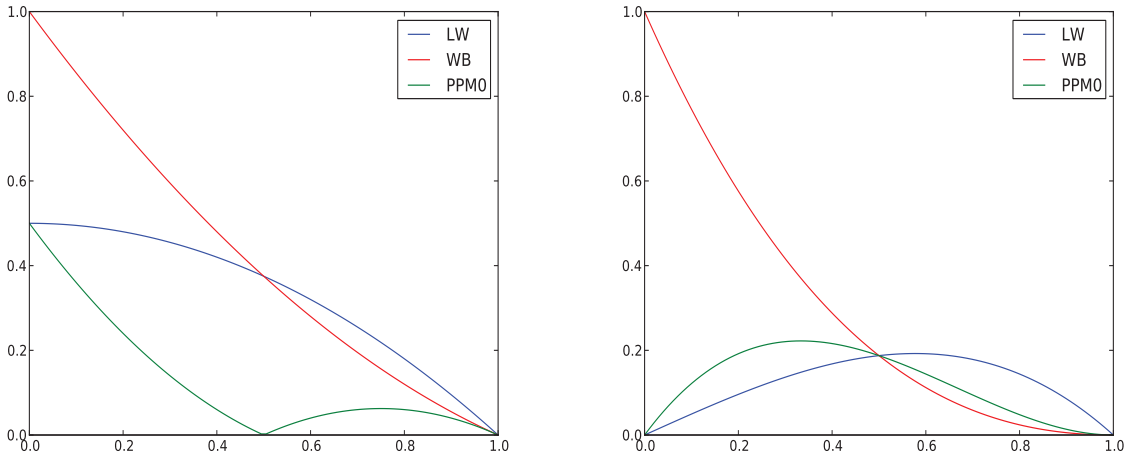


FIGURE 2.10 – Valeur absolue des coefficients de dispersion $|c_3|$ en fonction de η pour les schémas d'ordre 2 (à gauche) et valeur absolue des coefficients de dissipation $|c_4|$ en fonction de η pour les schémas d'ordre 2 (à droite).

Schéma d'ordre 3

	c_4	c_5
LAG 3	$(\eta^3-2\eta^2-\eta+2)/6$	$(2\eta^4-5\eta^3+5\eta-2)/12$
PPM1	$(\eta^3-2\eta^2+\eta)/6$	$(2\eta^4-5\eta^3+5\eta-2)/12$
PPM2	$(\eta^3-2\eta^2+\eta)/6$	$(2\eta^4-5\eta^3+4\eta^2-\eta)/12$
PSM	$(\eta^3-2\eta^2+\eta)/6$	$(6\eta^4-15\eta^3+10\eta^2-1)/36$

La Figure 2.11 présente les valeurs absolues des coefficients de dispersion $|c_4|$ et de dissipation $|c_5|$ en fonction de μ. Les coefficients c_4 et c_5 s'annulent pour $\eta = 1$: les schémas deviennent alors de simples translations résolvant exactement l'équation d'advection. De plus, le coefficient de dissipation $|c_4|$ est le même pour les schémas PPM1, PPM et PSM ; ce coefficient tend vers 0 lorsque η tend vers 0. Par contre, le schéma LAG3 peut être fortement

dissipatif pour de faibles valeurs de η. Pour le schéma PPM2, les coefficients de dissipation $|c_4|$ et de dispersion $|c_5|$ tendent vers 0 lorsque η tend vers 0.

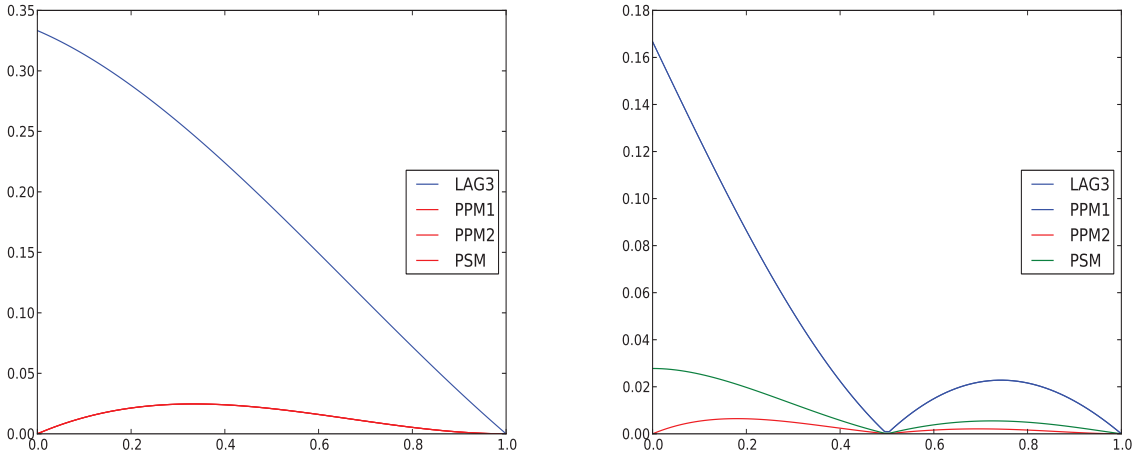


FIGURE 2.11 – Valeur absolue des coefficients de dissipation $|c_4|$ en fonction de η pour les schémas d'ordre 3 (à gauche) et valeur absolue des coefficients de dispersion $|c_5|$ en fonction de η pour les schémas d'ordre 3 (à droite).

2.3 Applications à un schéma avec limiteur de pente

Cette section quantifie de manière analytique la diffusion numérique d'un schéma de type Lagrange + Projection (L+P) [12] présenté dans la partie 2.3.1. Nous avons obtenu une formule exacte de la diffusion numérique de ce schéma au sens du terme de second ordre dans l'équation équivalente associée, pour une vitesse d'advection constante en temps mais dépendante de l'espace (voir la partie 2.3.2 pour les résultats et l'appendice F pour le détail des calculs).

Par diffusion numérique, nous entendons ici le terme de second ordre de l'équation équivalente associée au schéma (L+P). En effet, ce schéma est consistant, donc résout bien l'équation d'advection avec une erreur au moins d'ordre 1 :

$$\partial_t \rho + \partial_x(\rho u) = o(\Delta x) + o(\Delta t)$$

avec une densité $\rho(x,t)$ et une vitesse $u(x)$ constante en temps mais dépendante de l'espace. On définit le terme de diffusion numérique Π du schéma par :

$$\partial_t \rho + \partial_x(\rho u) = \Pi + o(\Delta x^2) + o(\Delta t^2).$$

Ce terme de diffusion dépend a priori du pas de temps Δt, du pas d'espace Δx et des gradients de la fonction ρ et de la vitesse u.

Dans la partie 2.3.3, nous avons validé ces formules en advection linéaire (vitesse constante en temps et en espace) par comparaison du résultat avec celui obtenu par une équation de diffusion, à la distance d'advection près.

2.3.1 Schéma Lagrange+Projection

L'équation d'étude est l'équation d'advection 1D suivante :

$$\partial_t \rho + \partial_x(u\rho) = 0,$$

où la densité $\rho(x,t)$ au point x au temps t est advectée par le champ de vitesse positif $u(x) \geq 0$ constant en temps.

Le schéma Eulérien (L+P) considéré dans cette partie se caractérise par une étape lagrangienne suivie d'une étape de projection. Il est d'ordre 2 en espace.

Le maillage est constitué de N mailles uniformes numérotées de 1 à N et de $N+1$ noeuds numérotés de $1/2$ à $N+1/2$. La maille j désigne donc le volume $1D$ entre les noeuds d'abscisses $x_{j-1/2}$ et $x_{j+1/2}$. Le pas d'espace $\Delta x = x_{j+1/2} - x_{j-1/2}$ et le pas de temps Δt sont ici supposés constants. Le champ de vitesse est défini aux noeuds $u_{j+1/2} = u(x_{j+1/2})$, la densité est définie aux mailles $\rho_j^n = \rho(x_j, t^n)$ où $x_j = (x_{j-1/2} + x_{j+1/2})/2$ et $t^n = n\Delta t$ ainsi que la masse $m_j^n = \rho_j^n \Delta x$.

Etape Lagrangienne

Lors de l'étape Lagrangienne, on avance les caractéristiques :

$$x_{j+1/2}^\ell = x_{j+1/2} + \Delta t u_{j+1/2}.$$

ce qui crée un nouveau maillage Lagrangien déformé dont les nœuds sont $(x_{j+1/2}^\ell)_j$. Les volumes des mailles deviennent alors $Vol_j^\ell = x_{j+1/2}^\ell - x_{j-1/2}^\ell$ et les centres des mailles sont notés $x_j^\ell = (x_{j-1/2}^\ell + x_{j+1/2}^\ell)/2$. La masse est conservée au cours de cette étape $m_j^\ell = m_j^n$, la densité des nouvelles mailles vaut $\rho_j^\ell = m_j^\ell/\Delta x$. Ces notations sont illustrées sur la Fig. 2.12.

Etape de projection

Les valeurs des densités calculées dans les mailles sur le maillage Lagrangien déformé vont être projetées sur le maillage initial dont les positions des nœuds sont $(x_{j+1/2})_j$. Les densités ρ_j^{n+1} obtenues après projection sur le maillage eulérien constituent la solution au temps t^{n+1} :

$$\rho_j^{n+1} = \rho_j^n - \frac{dm_{j+1/2} - dm_{j-1/2}}{\Delta x}$$

où le flux vaut :

$$dm_{j+1/2} = u_{j+1/2} \cdot \Delta t \cdot \rho_{j+1/2}^{n+1/2}.$$

Par définition,

$$\rho_{j+1/2}^{n+1/2} = \rho_j^\ell + \frac{\Delta \rho_j^\ell}{2}(\Delta x_j^\ell - u_{j+1/2} \cdot \Delta t)$$

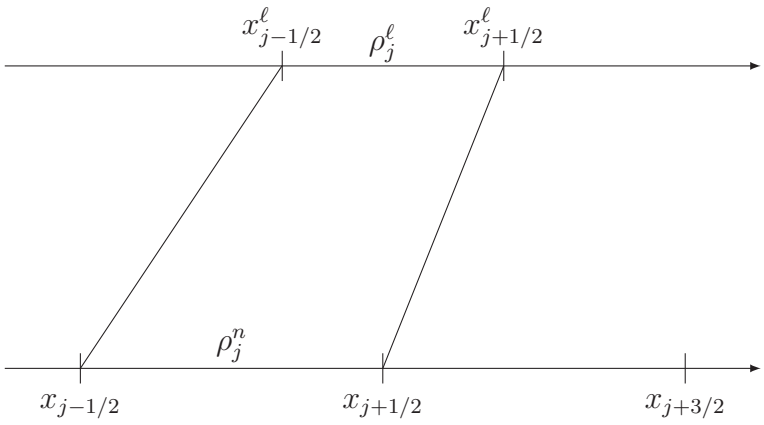


FIGURE 2.12 – Etape Lagrangienne.

où $\Delta x^\ell_j = x^\ell_{j+1/2} - x^\ell_{j-1/2}$ désigne la taille de la maille Lagrangienne et

$$\Delta\rho^\ell_j = \frac{1}{2}(\Psi^-_j \Delta\rho^\ell_{j-1/2} + \Psi^+_j \Delta\rho^\ell_{j+1/2})$$

est la variation de densité sur la maille Lagrangienne où la variation de densité sur les interfaces est définie par

$$\Delta\rho^\ell_{j+1/2} = \frac{\rho^\ell_{j+1} - \rho^\ell_i}{x^\ell_{j+1} - x^\ell_j}$$

et les limiteurs $\Psi^\pm$ sont calculés de la manière suivante :

$$\Psi^+_j = \Phi(r_j), \quad \Psi^-_j = \Phi\left(\frac{1}{r_j}\right)$$

avec

$$r_j = \frac{\Delta\rho^\ell_{j-1/2}}{\Delta\rho^\ell_{j+1/2}}, \quad \Phi(r) = \frac{r + |r|}{1 + r}.$$

Le rôle de la fonction $\Psi^\pm$ est de limiter les forts gradients et les gradients opposés : ainsi, lorsque $(\rho^\ell_{j+1} - \rho^\ell_j)/(\rho^\ell_j - \rho^\ell_{j-1}) < 0$, la fonction $\Psi^\pm$ s'annule ; lorsque $(\rho^\ell_{j+1} - \rho^\ell_j) >> (\rho^\ell_j - \rho^\ell_{j-1})$, on a $\Phi(r_j) \approx 2$ alors que $\Phi(1/r_j) \approx 0$.

Dans le cas contraire $r_j << 1$ (mais $r_j \geq 0$), $\varphi(y) \approx 0$ et $\Phi(1/y) = 2$, donc $\Delta\rho^\ell_j = (\rho^n_{j+1} - \rho^n_j)/\Delta x$.

D'autre part, lorsque la fonction est régulière (faibles gradients), $r_j \approx 1$ et $\varphi(r_j) \approx \varphi(1/r_j) \approx 2$, de sorte que

$$\Delta\rho^\ell_j = \frac{\rho^\ell_{j+1} - \rho^\ell_j}{x^\ell_{j+1} - x^\ell_j} + \frac{\rho^\ell_j - \rho^\ell_{j-1}}{x^\ell_j - x^\ell_{j-1}}$$

qui est alors simplement la dérivé d'ordre 2.

2.3.2 Résultats théoriques

Nous cherchons le terme de diffusion numérique Π défini par :

$$\partial_t \rho(x_i, t^n) + \partial_x(u\rho)(x_i, t^n) = \Pi(x_i, t^n) + o(\Delta x) + o(\Delta t).$$

Ce terme dépendra, au point x_i, des signes respectifs de r_{i-1} et de r_i. Nous considérerons, dans chacun de ces cas, l'advection constante et non constante. Nous donnons ci-dessous uniquement les résultats obtenus, les calculs seront détaillés dans l'Appendice F. Nous pouvons remarquer que dans le cas où $r_{i-1} \leq 0$ et $r_i \leq 0$, nous retrouvons le terme de diffusion du schéma Upwind. Dans le cas de l'advection à vitesse constante $u \equiv u_0$, nous noterons $\eta = \frac{u_0 \Delta t}{\Delta x}$.

- **Cas où $r_{i-1} \leq 0$ et $r_i \leq 0$ (pas de limiteur sur les cellules $i-1$ et i)**
  - *Advection constante :*
$$\Pi = \frac{\Delta x}{2} u_0 (1-\eta) \partial_{xx}\rho = \frac{\Delta x}{2} u_0 \partial_{xx}\rho - \frac{\Delta t}{2} u_0^2 \partial_{xx}\rho.$$
  - *Advection non constante :*
$$\Pi = \frac{\Delta x}{2}\Big((\partial_x u)(\partial_x \rho) + u\partial_{xx}\rho\Big) + \frac{\Delta t}{2}\Big(\rho(\partial_x u)^2 - u(\partial_x u)(\partial_x \rho) + \rho u \partial_{xx} u - u^2 \partial_{xx}\rho\Big).$$

- **Cas où $r_{i-1} \leq 0$ et $r_i > 0$**
  - *Advection constante :*
$$\Pi = u_0 \frac{\eta - 1}{2} \partial_x \rho + u_0 \Delta x \frac{(4-\eta)(\eta-1)^2}{8} \partial_{xx}\rho.$$
  - *Advection non constante :*
$$\begin{aligned}\Pi = {} & u \frac{\frac{\Delta t}{\Delta x} u - 1}{2} \partial_x \rho + \Delta t \Big(\frac{1}{2}(\partial_x u)^2 \rho + \frac{5}{8} u (\partial_x u)(\partial_x \rho) + \frac{3}{4} u \rho (\partial_{xx} u) - \frac{9}{8} u^2 (\partial_{xx}\rho) + \\ & \frac{13}{8} \frac{\Delta t}{\Delta x} u^2 (\partial_x \rho)(\partial_x u) + \frac{3}{4} \frac{\Delta t}{\Delta x} u^3 (\partial_{xx}\rho) + \frac{1}{4} \frac{\Delta t}{\Delta x} u^2 (\partial_{xx} u)\rho - \frac{1}{4} u^3 \left(\frac{\Delta t}{\Delta x}\right)^2 u^3 (\partial_x \rho)(\partial_x u) \\ & - \frac{1}{8}\left(\frac{\Delta t}{\Delta x}\right)^2 u^4 (\partial_{xx}\rho)\Big) + \Delta x \Big(\frac{1}{2} u (\partial_{xx}\rho) + \frac{1}{4}(\partial_x u)(\partial_x \rho)\Big).\end{aligned}$$

- **Cas où $r_{i-1} > 0$ et $r_i \leq 0$**
  - *Advection constante :*
$$\Pi = u_0 \frac{1-\eta}{2} \partial_x \rho + u_0 \Delta x \frac{\eta(1-\eta)(3+\eta)}{8} \partial_{xx}\rho.$$
  - *Advection non constante :*
$$\begin{aligned}\Pi = {} & u \frac{1 - \frac{\Delta t}{\Delta x} u}{2} \partial_x \rho + \Delta t \Big(\frac{1}{2}(\partial_x u)^2 \rho + \frac{3}{8} u (\partial_x u)(\partial_x \rho) + \frac{1}{4} u \rho (\partial_{xx} u) + \frac{3}{8} u^2 (\partial_{xx}\rho) \\ & - \frac{7}{8} \frac{\Delta t}{\Delta x} u^2 (\partial_x \rho)(\partial_x u) - \frac{1}{4} \frac{\Delta t}{\Delta x} u^3 (\partial_{xx}\rho) - \frac{1}{4} \frac{\Delta t}{\Delta x} u^2 (\partial_{xx} u)\rho - \frac{1}{4}\left(\frac{\Delta t}{\Delta x}\right)^2 u^3 (\partial_x \rho)(\partial_x u) \\ & - \frac{1}{8}\left(\frac{\Delta t}{\Delta x}\right)^2 u^4 (\partial_{xx}\rho)\Big) + \frac{\Delta x}{4}(\partial_x u)(\partial_x \rho).\end{aligned}$$

• **Cas où** $r_{i-1} > 0$ **et** $r_i > 0$

— *Advection constante*

$$\Pi = 0.$$

Nous cherchons alors le terme suivant $\tilde{\Pi}$ de l'équation équivalente

$$\tilde{\Pi} = u_0 \frac{\Delta x^2}{12}(2\eta^2 - 3\eta + 1)\partial_{xxx}\rho + u_0 \frac{\Delta x^3}{8}(\eta - 1)(\eta^2 - \eta + 1)\partial_{xxxx}\rho.$$

— *Advection non constante*

$$\Pi = \frac{\Delta t}{2}\Big(\rho(\partial_x u)^2 + u(\partial_x \rho)(\partial_x u) + u\rho(\partial_{xx} u)\Big).$$

2.3.3 Validation numérique en advection constante

Pour valider numériquement les coefficients de la partie précédente dans le cas de l'advection constante, nous discrétisons comme précédemment l'équation de diffusion associée par différences finies centrées. Par exemple, dans le cas sans limiteur ($r_{i-1} \leq 0$ et $r_i \leq 0$), l'équation de diffusion associée

$$\partial_t \rho = u_0 \frac{\Delta x}{2}(1 - \eta)\partial_{xx}\rho$$

est résolue à l'ordre 2 en espace par le schéma :

$$\frac{\rho_i^{n+1} - \rho_i^n}{\Delta t} = u_0 \frac{\Delta x}{2}(1 - \eta)\frac{\rho_{i+1}^n - 2\rho_i^n + \rho_{i-1}^n}{\Delta x^2}.$$

De même, l'ordre 4 est géré par

$$\partial_x^4 u(x_i, t_n) \approx \frac{u_{i+2} - 4u_{i+1} + 6u_i - 4u_{i-1} + u_{i-2}}{\Delta x^4}.$$

Nous avons appliqué le schéma avec et sans limiteur de pente à plusieurs profils de solution : une marche d'escalier (Figure 2.13), un Dirac discret (Figure 2.14), une parabole (Figure 2.15) et un double créneau (Figure 2.16). Dans tous les cas pour le schéma sans limiteur de pente, la solution de l'équation de diffusion est très proche du schéma. Avec limiteur de pente, la solution de l'équation de diffusion est proche du schéma bien que l'on observe des décrochages à certains endroits et plus particulièrement dans le cas du Dirac discret. Dans le cas général, on est non conservatif pour l'équation de diffusion.

2.3.4 Conclusion

L'approche par équation équivalente fonctionne. Les résultats semblent un peu moins bons pour le cas test Dirac. Il faut considérer le terme d'ordre 4 s'il n'y a pas de terme d'ordre 2 dans l'équation équivalente. Les calculs sont faits/faisables pour l'advection non constante en utilisant un logiciel de calcul formel.

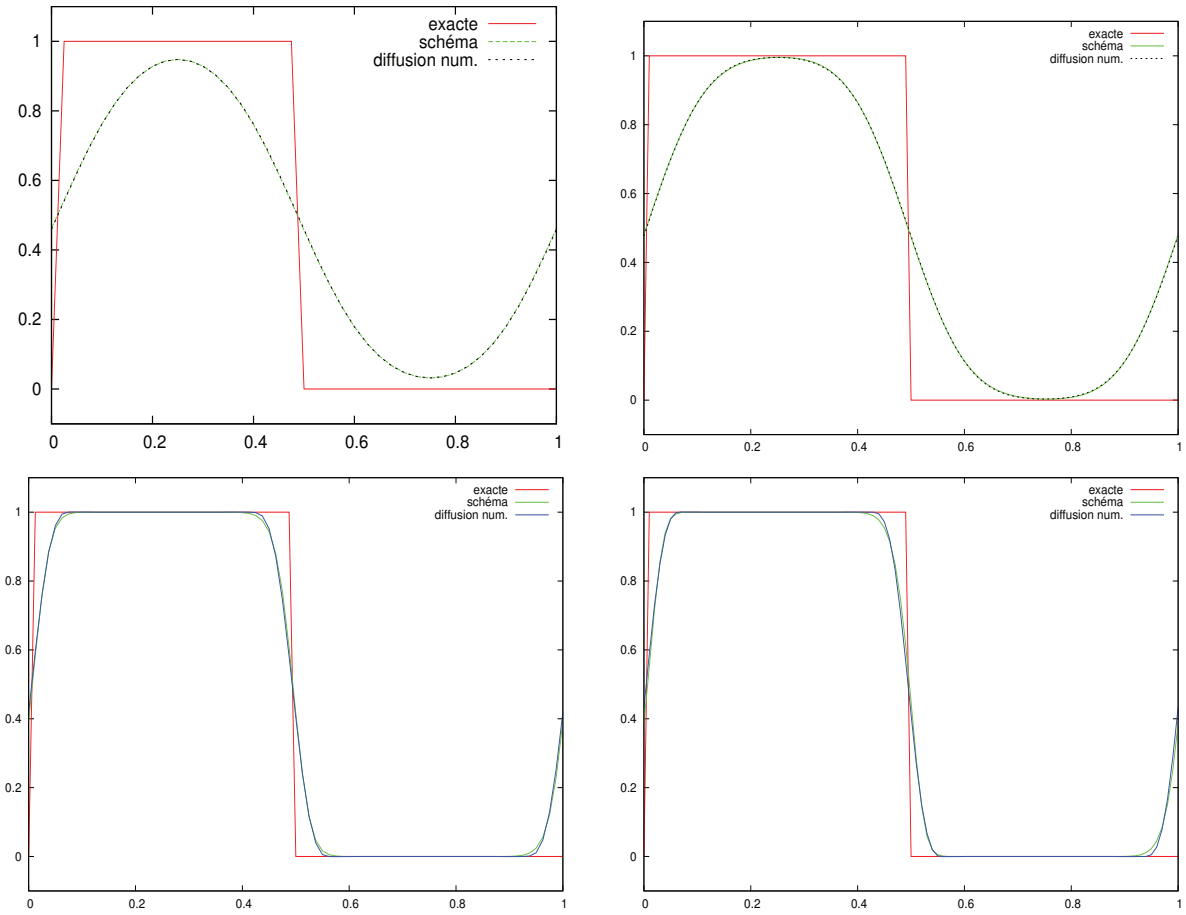


FIGURE 2.13 – Marche d'escalier, sans limiteur de pente, $\Delta x = 1/40$, $\Delta t = 0.01$, $\eta = 0.4$ (en haut à gauche), $\Delta x = 0.01$, $\Delta t = 0.0025$, $\eta = 0.25$ (en haut à droite) ; avec limiteur de pente, $\Delta x = 1/80$, $\Delta t = 0.005$, $\eta = 0.4$ (en bas à gauche), $\Delta x = 0.01$, $\Delta t = 0.0025$, $\eta = 0.25$ (en bas à droite).

En ce qui concerne les perspectives, il reste à effectuer une analyse mathématique et à comprendre pourquoi les termes non diffusifs de l'équation équivalente peuvent être négligés. De plus, il faut assurer la convergence des termes d'ordre élevé. Par ailleurs, il faudrait valider les résultats dans le cas de l'advection non constante. Enfin, on peut envisager une extension au multi-D.

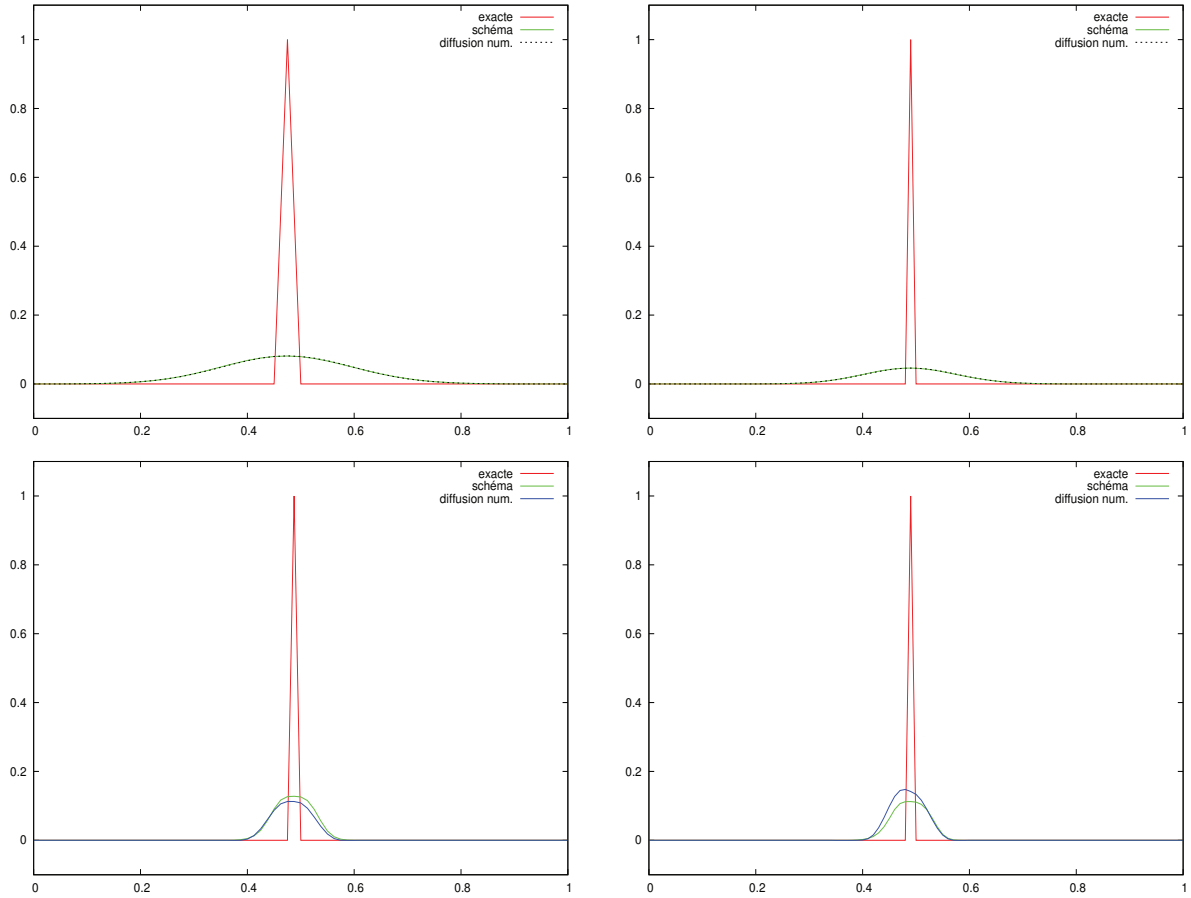


FIGURE 2.14 – Dirac discret, sans limiteur de pente, $\Delta x = 1/40$, $\Delta t = 0.01$, $\eta = 0.4$ (en haut à gauche), $\Delta x = 0.01$, $\Delta t = 0.0025$, $\eta = 0.25$ (en haut à droite) ; avec limiteur de pente, $\Delta x = 1/80$, $\Delta t = 0.005$, $\eta = 0.4$ (en bas à gauche), $\Delta x = 0.01$, $\Delta t = 0.0025$, $\eta = 0.25$ (en bas à droite).

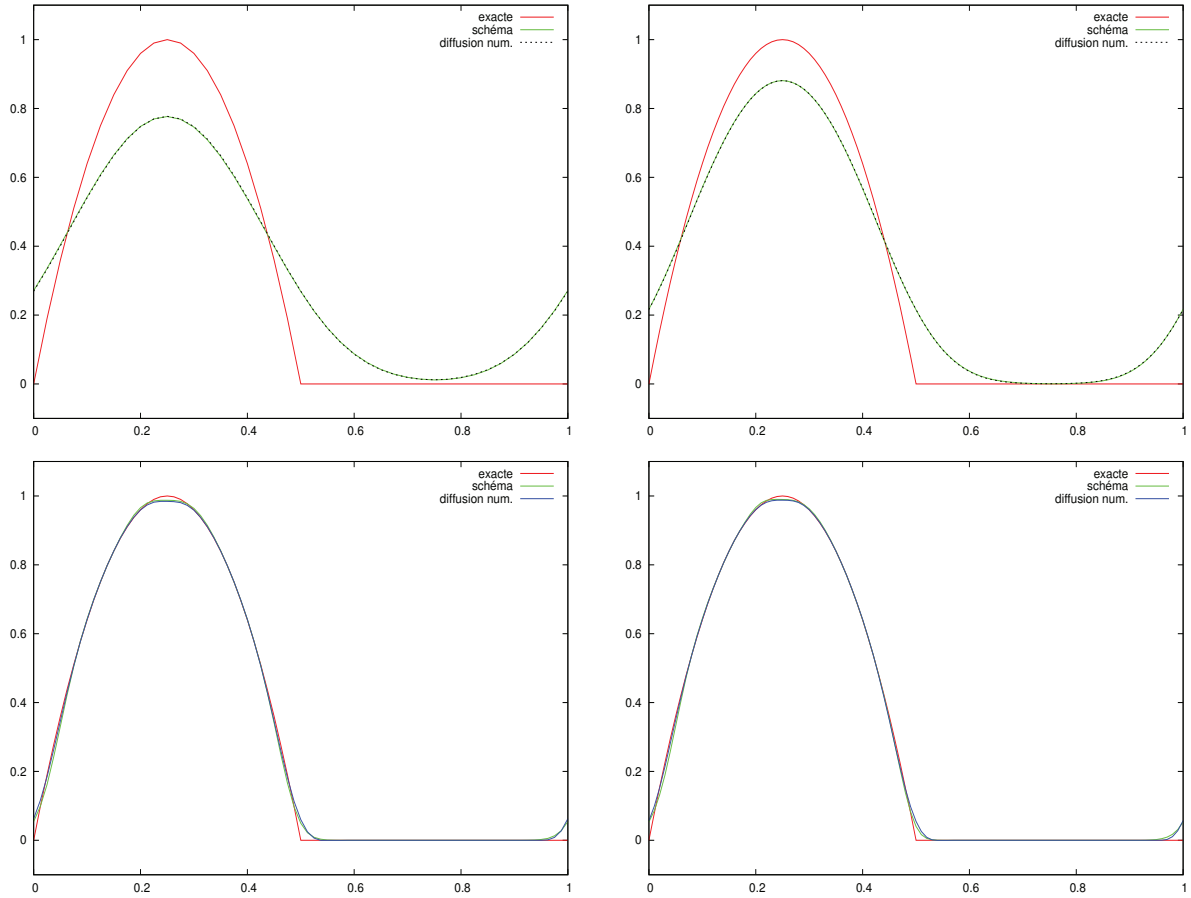


FIGURE 2.15 – Parabole, sans limiteur de pente, $\Delta x = 1/40$, $\Delta t = 0.01$, $\eta = 0.4$ (en haut à gauche), $\Delta x = 0.01$, $\Delta t = 0.0025$, $\eta = 0.25$ (en haut à droite) ; avec limiteur de pente, $\Delta x = 1/80$, $\Delta t = 0.005$, $\eta = 0.4$ (en bas à gauche), $\Delta x = 0.01$, $\Delta t = 0.0025$, $\eta = 0.25$ (en bas à droite).

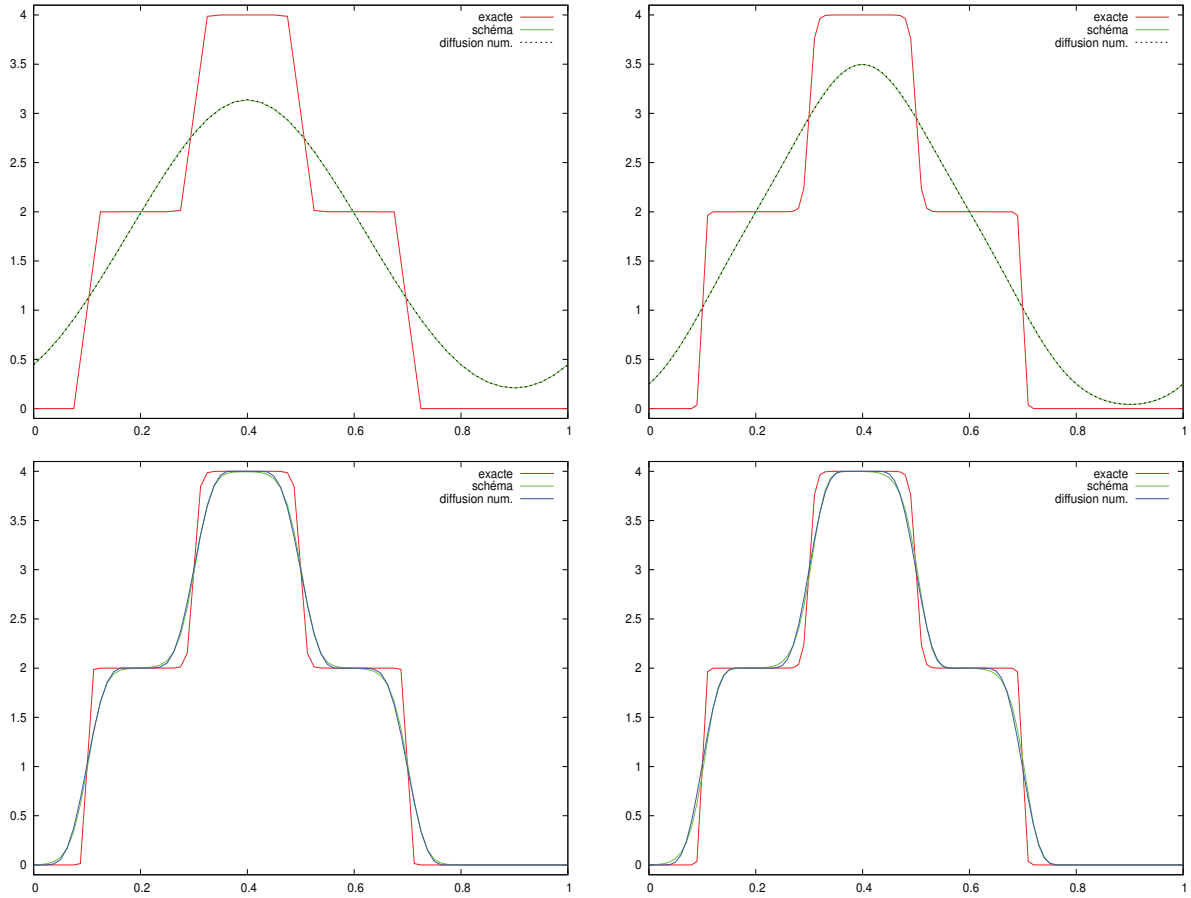


FIGURE 2.16 – Double créneau, sans limiteur de pente, $\Delta x = 1/40$, $\Delta t = 0.01$, $\eta = 0.4$ (en haut à gauche), $\Delta x = 0.01$, $\Delta t = 0.0025$, $\eta = 0.25$ (en haut à droite) ; avec limiteur de pente, $\Delta x = 1/80$, $\Delta t = 0.005$, $\eta = 0.4$ (en bas à gauche), $\Delta x = 0.01$, $\Delta t = 0.0025$, $\eta = 0.25$ (en bas à droite).

Chapitre 3

Superconvergence pour Galerkin Discontinu Semi-Lagrangien

Dans ce chapitre, nous discutons de la propriété de superconvengence pour le schéma Galerkin Discontinu Semi-Lagrangien (SLDG). Un tel schéma a déjà été développé dans [27] et plus récemment dans [28, 30, 25] en vue d'applications pour Vlasov-Maxwell/Poisson. Un point clé, dans de telles applications, est d'utiliser les splittings directionnels qui conduisent à une succession de problèmes d'advection constante et le schéma a l'avantage de ne pas être restreint par une condition CFL. Le cas de l'advection non constante est plus délicat et peut conduire à des stratégies différentes pour l'évaluation du flux qui devra être évalué ; voir [28] pour une discussion, et [29] pour un travail pionnier sur le sujet dans un cadre général. Le schéma SLDG continue à être étudié, voir [22, 21, 73].
La superconvergence des méthodes de type Galerkin discontinu a fait l'objet de nombreux développements. Dans [26], une approche par Fourier est utilisée pour analyser les propriétés de superconvergence. Bien qu'elle soit limitée aux maillages uniformes et périodiques dans le cadre de problèmes linéaires, l'approche par Fourier permet de donner des informations précises sur l'erreur. Cependant, cette approche est restreinte aux petits degrés puisque les calculs formels deviennent de plus en plus complexes à fur et à mesure que le degré augmente. D'autres techniques ont été développées pour traiter des cas plus généraux, comme le post-processing introduit dans [24] ; de noubreuses autres références se trouvent dans [26].
Nous considérons ici la superconvergence su schéma SLDG dans le cas de l'équation d'advection linéaire à vitesse constante avec des conditions aux bords périodiques. Nous énonçons un résultat pour un degré *quelconque*, ce qui ne semble pas encore avoir été considéré pour le schéma SLDG.

Nous montrons formellement et numériquement ce résultat pour de petits degrés. Nous donnons des pistes pour une démonstration générale pour tout degré (qui n'est pas faite ici) en utilisant une décomposition vectorielle dans l'espace de Fourier, l'inégalité de Cauchy-Schwarz et la formule d'Euler-MacLaurin.
La partie est organisée comme suit. Dans la partie 1, nous introduisons le schéma Galerkin discontinu semi-lagrangien. L'énoncé du théorème de superconvergence est donné dans la partie 2. Dans la partie 3, nous décrivons la structure de l'erreur de troncature et de l'erreur numérique. Une analyse de la structure propre de la matrice d'amplification du schéma est effectuée dans la partie 4. Dans la partie 5, nous rappelons les résultats précédents et discutons de la preuve générale. Les résultats formels et numériques montrant la superconvergence

sont présentés dans la partie 6.

3.1 Schéma Galerkin Discontinu Semi-Lagrangien

3.1.1 Notations

L'équation d'étude est l'équation d'advection linéaire

$$\begin{array}{ll} \partial_t f + a\partial_x f = 0 & (x,t) \in [0,1] \times [0,+\infty[\\ f(0,x) = f_0(x) & x \in [0,1] \end{array}$$

avec une vitesse constante $a > 0$.

Soit le domaine $\Omega = [0,1]$ qui est divisé en N cellules :

$$C_i = [x_{i-1/2}, x_{i+1/2}], \quad i = 0, ..., N-1.$$

Nous supposons ici que le maillage est uniforme : le pas d'espace Δx satisfait

$$\Delta x = x_{i+1/2} - x_{i-1/2} = \frac{1}{N}, \qquad i = 0, ..., N-1.$$

Nous définissons également le pas de temps Δt qui est également supposé être constant et nous notons $t^n = n\Delta t$. Des conditions périodiques aux bords sont utilisées.

3.1.2 Schéma SLDG

Soit $d \in \mathbb{N}$. Sur chaque cellule $C_i = [x_{i-1/2}, x_{i+1/2}]$, nous considérons $d+1$ points de Gauss désignés par $\{x_{ij}\}_{(i,j)\in\{0,...,N-1\}\times\{0,...,d\}}$. Notons par $\{\alpha_j\}_{j\in\{0,...,d\}}$ les points de Gauss dans l'intervalle $[0,1]$ et $\{\omega_j\}_{j\in\{0,...,d\}}$ leurs poids associés, nous introduisons d'abord les polynômes de Lagrange aux points α_j restreints à l'intervalle $[0,1]$:

$$\varphi_j(x) = \prod_{\ell,\ell\neq j} \frac{x-\alpha_\ell}{\alpha_j - \alpha_\ell} \text{ for } x \in [0,1], \qquad \varphi_j(x) = 0 \text{ sinon}$$

et les polynômes correspondants sur la cellule C_i :

$$\varphi_{ij}(x) = \varphi_j\left(\frac{x - x_{i-1/2}}{\Delta x}\right).$$

En écrivant $f^n \approx f(t^n, \cdot)$ sous la forme $f^n(x) = \sum_{i,j} f_{ij}^n \varphi_{ij}(x)$, les degrés de liberté $f_{ij}^n \approx f(t^n, \alpha_{ij})$ sont donnés par

$$\omega_j \Delta x f_{ij}^n = \int_{\mathbb{R}} f^n(x)\varphi_{ij}(x)dx.$$

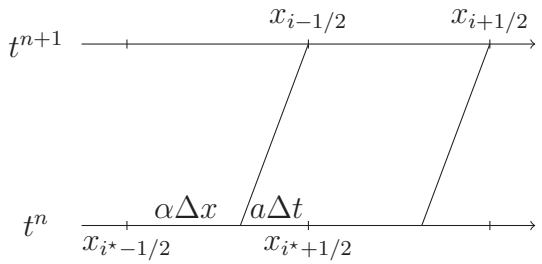
En utilisant l'équation d'advection pour mettre à jour les degrés de liberté, on obtient le schéma Galerkin Discontinu semi-lagrangien (SLDG) :

$$\omega_j \Delta x f_{ij}^{n+1} = \int_{\mathbb{R}} f^n(x - a\Delta t)\varphi_{ij}(x)dx.$$

Ceci conduit à

$$\omega_j \Delta x f_{ij}^{n+1} = \sum_{k=0}^{N-1} \sum_{\ell=0}^{d} f_{k\ell}^n \int_{\mathbb{R}} \varphi_\ell \left(\frac{x - a\Delta t - x_{k-1/2}}{\Delta x} \right) \varphi_j \left(\frac{x - x_{i-1/2}}{\Delta x} \right) dx.$$

En définissant $i^\star$ et α tels que $x_{i-1/2} - a\Delta t = x_{i^\star - 1/2} + \alpha \Delta x$,



et en utilisant le changement de variable $x = x_{i-1/2} + s\Delta x$, on obtient :

$$\omega_j \Delta x f_{ij}^{n+1} = \Delta x \sum_{k=0}^{N-1} \sum_{\ell=0}^{d} f_{k\ell}^n \int_{\mathbb{R}} \varphi_\ell(i^\star - k + \alpha + s) \varphi_j(s) ds$$

et finalement, on obtient la formule explicite suivante pour le schéma SLDG :

$$\omega_j f_{ij}^{n+1} = \sum_{\ell=0}^{d} f_{i^\star,\ell}^n \int_{\mathbb{R}} \varphi_\ell(\alpha + s) \varphi_j(s) ds + \sum_{\ell=0}^{d} f_{i^\star+1,\ell}^n \int_{\mathbb{R}} \varphi_\ell(\alpha + s - 1) \varphi_j(s) ds. \tag{3.1.1}$$

Nous définissons la norme L^2 discrète comme suit :

$$\|z\|_2^2 = \frac{1}{N} \sum_{i=0}^{N-1} \sum_{j=0}^{d} \omega_j z_{i,j}^2, \ z = (z_{i,j}) \in \mathbb{R}^{N(d+1)}.$$

3.2 Propriété de superconvergence

Ce chapitre concerne la conjecture de superconvergence suivante.

Conjecture 3.2.1. *Considérons l'équation d'advection linéaire à vitesse constante*

$$\begin{aligned} &\partial_t f + a\partial_x f = 0 && (x,t) \in [0,1] \times [0,T] \\ &f(0,x) = f^0(x) && x \in [0,1] \end{aligned}$$

avec $f^0 \in \mathcal{C}^{2d+2}([0,1])$ et le schéma Galerkin Discontinu Semi-Lagrangien pour la discrétisation de cette équation. Nous écrivons $-\frac{a\Delta t}{\Delta x} = i_0^\star + \alpha$ où $i_0^\star \in \mathbb{Z}$ et $0 \leq \alpha < 1$. Alors il existe des constantes $C_1, C_2 > 0$ dépendantes de α, de la régularité de la solution et indépendantes du temps T telles que l'erreur numérique au point x_{ij} : $e_{ij}^n = f_{ij}^n - f(t^n, x_{ij})$ est bornée en norme discrète L^2 :

$$\|\boldsymbol{e}^n\|_2 \leq C_1 \Delta x^{d+1} + n C_2 \Delta x^{2d+2}.$$

Remarque 3.2.2. *Cette conjecture sera prouvée pour d=1. Le cas d=2 peut être traité formellement pour une valeur de α donné ($\alpha = 0.27$ par exemple dans la partie 3.6.1). Les résultats numériques de partie 3.6.2 viennent confirmer ce résultat.*

Remarque 3.2.3. *Nous pouvons nous restreindre au cas où $0 < \frac{a\Delta t}{\Delta x} < 1$ en utilisant le fait que le schéma est exact quand $\frac{a\Delta t}{\Delta x} \in \mathbb{Z}$.*

Remarque 3.2.4. *Un avantage de cette nouvelle estimation est d'avoir une convergence sur un temps long. En effet, si $1 \leq \beta \leq d$ et si nous fixons les valeurs de Δx et Δt, nous cherchons le plus grand temps $T = n\Delta t$ tel que*

$$\|\boldsymbol{e}^n\|_2 \leq C_4 \Delta x^\beta$$

où C_4 est une constante.

1. *Dans le cas classique, nous avons $\|\boldsymbol{e}^n\|_2 \leq C_5 n \Delta x^{d+1}$ ce qui conduit à*

$$T \leq C_6 \Delta x^{\beta - d}$$

   où C_5, C_6 sont des constantes.

2. *Dans le cas de la superconvergence, nous avons*

$$T \leq C_7 \Delta x^{\beta - 2d - 1}$$

   où C_7 est une constante.

Remarque 3.2.5. *Le schéma considéré ici est différent de celui décrit dans [26]. Le schéma dans [26] correspond à l'intégrateur exponentiel quand $\alpha \to 0$ du schéma considéré dans ce chapitre, ce qui n'est pas considéré dans cette analyse, mais peut être adapté. Voir [23] pour un exemple similaire.*

Remarque 3.2.6. *Le degré est ici arbitraire. L'approche de Fourier, comme dans [26], est possible pour les petits degrés (voir Section 7). Cela donne des informations plus précises mais la complexité de calcul augmente fortement avec le degré, et semble devenir impossible pour des degrés plus grands.*

3.3 Erreur de troncature et erreur numérique

3.3.1 Erreur de troncature

Nous notons $\mathbf{x} = (x_{ij})_{i=0..N-1, j=0..d}$.

Notation 3.3.1. *$\mathcal{S} : \mathbb{R}^{N(d+1)} \to \mathbb{R}^{N(d+1)}$ est l'opérateur du schéma Galerkin Discontinu Semi-Lagrangien. donné par (3.1.1).*

Notation 3.3.2. *L'erreur de troncature au point x_{ij} et au temps t^n est défini par*

$$g_{ij}^n = \frac{1}{\Delta t} \left(f(t^{n+1}, x_{ij}) - \mathcal{S}(f(t^n, \boldsymbol{x}))_{ij} \right).$$

La proposition suivante donne une expression de l'erreur de troncature.

Proposition 3.3.3. *Il existe des constantes E_j^k independantes de i et n et dépendantes de α telles que*

$$g_{ij}^n = \sum_{k=0}^{2d+1} E_j^k \frac{\Delta x^k}{\Delta t} \partial_x^k f(t^n, x_{ij}) + \mathcal{O}\left(\frac{\Delta x^{2d+2}}{\Delta t}\right).$$

Démonstration. Supposons que $0 < \frac{a\Delta t}{\Delta x} < 1$, alors $i^\star = i - 1$ et $\alpha = 1 - \frac{a\Delta t}{\Delta x}$. Le calcul de l'erreur de troncature au point x_{ij} défini par

$$\frac{1}{\Delta t}\left(f(t^{n+1}, x_{ij}) - \mathcal{S}(f(t^n, \mathbf{x}))_{ij}\right)$$

donne :

$$\frac{1}{\Delta t}\left(f(t^n + \Delta t, x_{ij}) - \frac{1}{\omega_j}\sum_{j'=0}^{d} f(t^n, (i - 1 + \alpha_{j'})\Delta x) \int_{s=\alpha}^{1} \varphi_{j'}(s)\varphi_j(s - \alpha)ds \right.$$
$$\left. -\frac{1}{\omega_j}\sum_{j'=0}^{d} f(t^n, (i + \alpha_{j'})\Delta x) \int_{s=0}^{\alpha} \varphi_{j'}(s)\varphi_j(s + 1 - \alpha)ds\right).$$

Nous utilisons l'équation d'advection constante pour revenir en arrière :

$$f(t^n + \Delta t, x_{ij}) = f(t^n, x_{ij} - a\Delta t)$$

et donc, par un développement de Taylor au point x_{ij}, l'erreur de troncature au point x_{ij} s'écrit

$$\sum_{k=0}^{2d+1} E_j^k \frac{\Delta x^k}{\Delta t} \partial_x^k f(t^n, x_{ij}) + \mathcal{O}\left(\frac{\Delta x^{2d+2}}{\Delta t}\right)$$

où

$$\begin{aligned} E_j^k &= \frac{(\alpha - 1)^k}{k!} - \frac{1}{\omega_j}\sum_{j'=0}^{d}\left(\frac{(\alpha_{j'} - \alpha_j - 1)^k}{k!}\int_{s=\alpha}^{1} \varphi_{j'}(s)\varphi_j(s - \alpha)ds + \right. \\ &\left. \frac{(\alpha_{j'} - \alpha_j)^k}{k!}\int_{s=0}^{\alpha} \varphi_{j'}(s)\varphi_j(s + 1 - \alpha)ds\right). \end{aligned}$$

□

Nous pouvons d'abord établir que les premiers coefficients sont nuls, ce qui conduit à l'estimation de convergence classique à l'ordre $d + 1$:

Proposition 3.3.4. *Pour tout $0 \leq k \leq d$ et tout $0 \leq j \leq d$, les coefficients de l'erreur de troncature, définis dans la proposition 3.3.3, satisfont*

$$E_j^k = 0.$$

Démonstration. Nous commençons par le cas $k = 0$:

$$\begin{aligned} E_j^0 &= 1 - \frac{1}{\omega_j} \sum_{j'=0}^{d} \left(\int_{s=\alpha}^{1} \varphi_{j'}(s)\varphi_j(s-\alpha)ds \; + \; \int_{s=0}^{\alpha} \varphi_{j'}(s)\varphi_j(s+1-\alpha)ds \right) \\ &= 1 - \frac{1}{\omega_j} \left[\int_{s=\alpha}^{1} \left(\sum_{j'=0}^{d} \varphi_{j'}(s) \right) \varphi_j(s-\alpha)ds + \int_{s=0}^{\alpha} \left(\sum_{j'=0}^{d} \varphi_{j'}(s) \right) \varphi_j(s+1-\alpha)ds \right]. \end{aligned}$$

Nous avons

$$1 - \sum_{j'=0}^{d} \varphi_{j'}(s) \equiv 0$$

puisque le terme de gauche est un polynôme de degré d avec $d+1$ zéros $(\alpha_0, ..., \alpha_d)$. Ainsi, nous obtenons

$$\begin{aligned} E_j^0 &= 1 - \frac{1}{\omega_j} \left(\int_{s=0}^{1-\alpha} \varphi_j(s)ds + \int_{s=1-\alpha}^{1} \varphi_j(s)ds \right) \\ &= 1 - \frac{1}{\omega_j} \int_{s=0}^{1} \varphi_j(s)ds \\ &= 1 - \sum_{i=0}^{d} \varphi_j(\alpha_i) \\ &= 1 - \sum_{i=0}^{d} \delta_{ij} \\ &= 0. \end{aligned}$$

Le coefficient E_j^k pour $1 \leq k \leq d$ s'écrit :

$$\begin{aligned} E_j^k = \; & \frac{(\alpha-1)^k}{k!} - \frac{1}{\omega_j} \sum_{j'=0}^{d} \left(\frac{(\alpha_{j'} - \alpha_j - 1)^k}{k!} \int_{s=\alpha}^{1} \varphi_{j'}(s)\varphi_j(s-\alpha)ds + \right. \\ & \left. \frac{(\alpha_{j'} - \alpha_j)^k}{k!} \int_{s=0}^{\alpha} \varphi_{j'}(s)\varphi_j(s+1-\alpha)ds \right). \end{aligned}$$

Pour $1 \leq k \leq d$,

$$\sum_{j'=0}^{d} (\alpha_{j'} - \alpha_j - 1)^k \varphi_{j'}(s) \equiv (s - \alpha_j - 1)^k$$

et

$$\sum_{j'=0}^{d} (\alpha_{j'} - \alpha_j)^k \varphi_{j'}(s) \equiv (s - \alpha_j)^k$$

puisque les termes de droite et de gauche sont des polynômes de degré au plus d avec $d+1$ valeurs communes en α_i $(i = 0, \ldots, d)$. Alors : car les termes à droite et à gauche sont des

polynômes de degré au plus d avec $d+1$ valeurs communes aux points α_i $(i = 0, \ldots, d)$. Alors :

$$\begin{aligned} E_j^k &= \frac{1}{k!}\Bigg[(\alpha-1)^k - \frac{1}{\omega_j}\int_{s=\alpha}^{1}(s-\alpha_j-1)^k\varphi_j(s-\alpha)ds + \\ &\quad -\frac{1}{\omega_j}\int_{s=0}^{\alpha}(s-\alpha_j)^k\varphi_j(s+1-\alpha)ds\Bigg]. \end{aligned}$$

Par changement de variable, nous obtenons

$$E_j^k = \frac{1}{k!}\Bigg[(\alpha-1)^k - \frac{1}{\omega_j}\int_{s=0}^{1}(t-1+\alpha-\alpha_j)^k\varphi_j(t)dt\Bigg].$$

Le polynôme

$$t \mapsto (t-1+\alpha-\alpha_j)^k\varphi_j(t)$$

est de degré inférieur ou égal à $2d$, donc, par la formule de quadrature de Gauss, on obtient :

$$E_j^k = \frac{1}{k!}\Bigg[(\alpha-1)^k - (\alpha_j-1+\alpha-\alpha_j)^k\Bigg] = 0.$$

□

Dans le but d'obtenir la propriété de superconvergence, nous avons la plus faible propriété suivante qui est valide également pour les termes d'ordre plus élevés, jusqu'au degré $2d+1$:

Proposition 3.3.5. *Pour tout $k = 0, \ldots, 2d+1$, les coefficients de l'erreur de troncature, definis dans la Proposition 3.3.3, satisfont*

$$\sum_{j=0}^{d}\omega_j E_j^k = 0.$$

Démonstration. Notons que les cas $k = 0, \ldots, d$ sont déjà donnés par la proposition précédente. Le coefficient E_j^k s'écrit

$$\begin{aligned} E_j^k &= \frac{(\alpha-1)^k}{k!} - \frac{1}{\omega_j}\sum_{j'=0}^{d}\Bigg(\frac{(\alpha_{j'}-\alpha_j-1)^k}{k!}\int_{s=\alpha}^{1}\varphi_{j'}(s)\varphi_j(s-\alpha)ds + \\ &\quad \frac{(\alpha_{j'}-\alpha_j)^k}{k!}\int_{s=0}^{\alpha}\varphi_{j'}(s)\varphi_j(s+1-\alpha)ds\Bigg) \end{aligned}$$

d'où

$$\begin{aligned} \sum_{j=0}^{d}\omega_j E_j^k &= \frac{1}{k!}\Bigg((\alpha-1)^k - \sum_{j=0}^{d}\sum_{j'=0}^{d}\Bigg((\alpha_{j'}-\alpha_j-1)^k\int_{s=\alpha}^{1}\varphi_{j'}(s)\varphi_j(s-\alpha)ds + \\ &\quad (\alpha_{j'}-\alpha_j)^k\int_{s=0}^{\alpha}\varphi_{j'}(s)\varphi_j(s+1-\alpha)ds\Bigg)\Bigg). \end{aligned}$$

Nous avons donc à établir que pour tout $d+1 \leq k \leq 2d+1$:

$$\sum_{j=0}^{d}\sum_{j'=0}^{d}\left((\alpha_{j'}-\alpha_j-1)^k\int_{s=\alpha}^{1}\varphi_{j'}(s)\varphi_j(s-\alpha)ds+(\alpha_{j'}-\alpha_j)^k\int_{s=0}^{\alpha}\varphi_{j'}(s)\varphi_j(s+1-\alpha)ds\right)=(\alpha-1)^k.$$

Dans cette preuve, nous utilisons les propriétés suivantes :

(P1) Pour tout polynôme P de degré inférieur ou égal à d, nous avons

$$\sum_{j=0}^{d}P(\alpha_j)\varphi_j(s)\equiv P(s).$$

(P2) Pour tout polynôme P de degré inférieur ou égal à $2d+1$, nous avons

$$\int_0^1 P(x)dx=\sum_{i=0}^{d}\omega_i P(\alpha_i).$$

En utilisant le binôme de Newton et en séparant les cas $r \leq d$ et $r > d$, nous obtenons :

$$\begin{aligned}
A &:= \sum_{j=0}^{d}\sum_{j'=0}^{d}\left((\alpha_{j'}-\alpha_j-1)^k\int_{s=\alpha}^{1}\varphi_{j'}(s)\varphi_j(s-\alpha)ds+(\alpha_{j'}-\alpha_j)^k\int_{s=0}^{\alpha}\varphi_{j'}(s)\varphi_j(s+1-\alpha)ds\right)\\
&= \sum_{j,j'=0}^{d}\sum_{r=0}^{d}\int_{s=\alpha}^{1}\binom{k}{r}\alpha_{j'}^r(-\alpha_j-1)^{k-r}\varphi_{j'}(s)\varphi_j(s-\alpha)ds\\
&\quad+\sum_{j,j'=0}^{d}\sum_{r=d+1}^{k}\int_{s=\alpha}^{1}\binom{k}{r}\alpha_{j'}^r(-\alpha_j-1)^{k-r}\varphi_{j'}(s)\varphi_j(s-\alpha)ds\\
&\quad+\sum_{j,j'=0}^{d}\sum_{r=0}^{d}\int_{s=0}^{\alpha}\binom{k}{r}\alpha_{j'}^r(-\alpha_j)^{k-r}\varphi_{j'}(s)\varphi_j(s+1-\alpha)ds\\
&\quad+\sum_{j,j'=0}^{d}\sum_{r=d+1}^{k}\int_{s=0}^{\alpha}\binom{k}{r}\alpha_{j'}^r(-\alpha_j)^{k-r}\varphi_{j'}(s)\varphi_j(s+1-\alpha)ds.
\end{aligned}$$

Puisque $k \leq 2d+1$, la relation $r > d$ implique $k-r \leq d$. Nous pouvons utiliser la propriété

(P1) dans chaque terme :

$$\begin{aligned} A &= \sum_{j=0}^{d}\sum_{r=0}^{d}\int_{s=\alpha}^{1}\binom{k}{r}s^r(-\alpha_j-1)^{k-r}\varphi_j(s-\alpha)ds \\ &\quad+\sum_{j'=0}^{d}\sum_{r=d+1}^{k}\int_{s=\alpha}^{1}\binom{k}{r}\alpha_{j'}^r(-s+\alpha-1)^{k-r}\varphi_{j'}(s)ds \\ &\quad+\sum_{j=0}^{d}\sum_{r=0}^{d}\int_{s=0}^{\alpha}\binom{k}{r}s^r(-\alpha_j)^{k-r}\varphi_j(s+1-\alpha)ds \\ &\quad+\sum_{j'=0}^{d}\sum_{r=d+1}^{k}\int_{s=0}^{\alpha}\binom{k}{r}\alpha_{j'}^r(-s+\alpha-1)^{k-r}\varphi_{j'}(s)ds \\ &=: (1)+(2)+(3)+(4). \end{aligned}$$

Nous calculons d'abord la somme (2) + (4) puisque seules les bornes d'intégration diffèrent dans les 2 expressions.

$$(2)+(4) = \sum_{j'=0}^{d}\sum_{r=d+1}^{k}\int_{s=0}^{1}\binom{k}{r}\alpha_{j'}^r(-s+\alpha-1)^{k-r}\varphi_{j'}(s)ds.$$

Comme $s\mapsto(-s+\alpha-1)^{k-r}\varphi_{j'}(s)$ est un polynôme de degré inférieur ou égal à $2d$, nous pouvons appliquer la propriété (P2) :

$$(2)+(4) = \sum_{j'=0}^{d}\sum_{r=d+1}^{k}\binom{k}{r}\alpha_{j'}^r(-\alpha_{j'}+\alpha-1)^{k-r}\omega_{j'}.$$

En appliquant la propriété (P2), cette fois au polynôme $s\mapsto s^r(-s+\alpha-1)^{k-r}$ qui est de degré $2d+1$:

$$(2)+(4) = \sum_{r=d+1}^{k}\binom{k}{r}\int_0^1 s^r(-s+\alpha-1)^{k-r}\varphi_{j'}(s)ds.$$

Passons au calcul de (1)+(3). En faisant le changement de variables $s=s-\alpha$ et $s=s+1-\alpha$, nous obtenons

$$\begin{aligned} (1)+(3) &= \sum_{j=0}^{d}\sum_{r=0}^{d}\int_{s=0}^{1-\alpha}\binom{k}{r}(s+\alpha)^r(-\alpha_j-1)^{k-r}\varphi_j(s)ds \\ &\quad+\sum_{j=0}^{d}\sum_{r=0}^{d}\int_{s=1-\alpha}^{1}\binom{k}{r}(s-1+\alpha)^r(-\alpha_j)^{k-r}\varphi_j(s)ds. \end{aligned}$$

En séparant de nouveau les cas $r \leq d$ et $r > d$:

$$\begin{aligned}
(1)+(3) &= \sum_{j=0}^{d}\sum_{r=0}^{k}\int_{s=0}^{1-\alpha}\binom{k}{r}(s+\alpha)^r(-\alpha_j-1)^{k-r}\varphi_j(s)ds \\
&\quad -\sum_{j=0}^{d}\sum_{r=d+1}^{k}\int_{s=0}^{1-\alpha}\binom{k}{r}(s+\alpha)^r(-\alpha_j-1)^{k-r}\varphi_j(s)ds \\
&\quad +\sum_{j=0}^{d}\sum_{r=0}^{k}\int_{s=1-\alpha}^{1}\binom{k}{r}(s-1+\alpha)^r(-\alpha_j)^{k-r}\varphi_j(s)ds \\
&\quad -\sum_{j=0}^{d}\sum_{r=d+1}^{k}\int_{s=1-\alpha}^{1}\binom{k}{r}(s-1+\alpha)^r(-\alpha_j)^{k-r}\varphi_j(s)ds.
\end{aligned}$$

Nous pouvons alors utiliser le binôme de Newton et la propriété (P1) pour les polynômes $(-s-1)^{k-r}$ et $(-s)^{k-r}$:

$$\begin{aligned}
(1)+(3) &= \sum_{j=0}^{d}\int_{s=0}^{1-\alpha}(s+\alpha-\alpha_j-1)^k\varphi_j(s)ds \\
&\quad -\sum_{j=0}^{d}\sum_{r=d+1}^{k}\int_{s=0}^{1-\alpha}\binom{k}{r}(s+\alpha)^r(-\alpha_j-1)^{k-r}\varphi_j(s)ds \\
&\quad +\sum_{j=0}^{d}\int_{s=1-\alpha}^{1}(s-1+\alpha-\alpha_j)^k\varphi_j(s)ds \\
&\quad -\sum_{j=0}^{d}\sum_{r=d+1}^{k}\int_{s=1-\alpha}^{1}\binom{k}{r}(s-1+\alpha)^r(-\alpha_j)^{k-r}\varphi_j(s)ds \\
&= \sum_{j=0}^{d}\int_{s=0}^{1}(s+\alpha-\alpha_j-1)^k\varphi_j(s)ds \\
&\quad -\sum_{r=d+1}^{k}\int_{s=0}^{1-\alpha}\binom{k}{r}(s+\alpha)^r(-s-1)^{k-r}ds \\
&\quad -\sum_{r=d+1}^{k}\int_{s=1-\alpha}^{1}\binom{k}{r}(s-1+\alpha)^r(-s)^{k-r}ds.
\end{aligned}$$

Nous faisons le changement de variables $s = s - 1$ dans le terme précédent :

$$\begin{aligned}(1)+(3) &= \sum_{j=0}^{d}\int_{s=0}^{1}(s+\alpha-\alpha_j-1)^k\varphi_j(s)ds \\ &\quad -\sum_{r=d+1}^{k}\int_{s=0}^{1-\alpha}\binom{k}{r}(s+\alpha)^r(-s-1)^{k-r}ds \\ &\quad -\sum_{r=d+1}^{k}\int_{s=-\alpha}^{0}\binom{k}{r}(s+\alpha)^r(-s-1)^{k-r}ds \\ &= \sum_{j=0}^{d}\int_{s=0}^{1}(s+\alpha-\alpha_j-1)^k\varphi_j(s)ds \\ &\quad -\sum_{r=d+1}^{k}\int_{s=-\alpha}^{1-\alpha}\binom{k}{r}(s+\alpha)^r(-s-1)^{k-r}ds.\end{aligned}$$

Le changement de variable $s = s + \alpha$ dans le terme précédent conduit à :

$$\begin{aligned}(1)+(3) &= \sum_{j=0}^{d}\int_{s=0}^{1}(s+\alpha-\alpha_j-1)^k\varphi_j(s)ds \\ &\quad -\sum_{r=d+1}^{k}\int_{s=0}^{1}\binom{k}{r}s^r(-s+\alpha-1)^{k-r}ds.\end{aligned}$$

Nous sommons finalement nos deux résultats intermédiaires et nous obtenons :

$$\begin{aligned}(1)+(2)+(3)+(4) &= \sum_{j=0}^{d}\int_{s=0}^{1}(s+\alpha-\alpha_j-1)^k\varphi_j(s)ds \\ &= \sum_{j=0}^{d}\sum_{r=0}^{k}\binom{k}{r}\int_{s=0}^{1}(s+\alpha)^r(-\alpha_j-1)^{k-r}\varphi_j(s)ds.\end{aligned}$$

En séparant les cas $r \leq d$ et $r > d$:

$$\begin{aligned}(1)+(2)+(3)+(4) &= \sum_{j=0}^{d}\sum_{r=0}^{d}\binom{k}{r}\int_{s=0}^{1}(s+\alpha)^r(-\alpha_j-1)^{k-r}\varphi_j(s)ds \\ &\quad +\sum_{j=0}^{d}\sum_{r=d+1}^{k}\binom{k}{r}\int_{s=0}^{1}(s+\alpha)^r(-\alpha_j-1)^{k-r}\varphi_j(s)ds.\end{aligned}$$

Pour le premier terme, nous utilisons la propriété (P2) avec le polynôme $s \mapsto (s+\alpha)^r\varphi_j(s)$ de degré maximal $2d$. Dans le second terme, nous utilisons la propriété (P1) appliquée au

polynôme $s \mapsto (-s-1)^{k-r}$ de degré maximal d :

$$\begin{aligned}(1)+(2)+(3)+(4) &= \sum_{j=0}^{d}\sum_{r=0}^{d}\binom{k}{r}\omega_j(\alpha_j+\alpha)^r(-\alpha_j-1)^{k-r} \\ &\quad + \sum_{r=d+1}^{k}\binom{k}{r}\int_{s=0}^{1}(s+\alpha)^r(-s-1)^{k-r}ds.\end{aligned}$$

Puisque le polynôme $s \mapsto (s+\alpha)^r(-s-1)^{k-r}$ est de degré $k \leq 2d+1$, nous pouvons appliquer (P2) dans le dernier terme :

$$\begin{aligned}(1)+(2)+(3)+(4) &= \sum_{r=0}^{d}\binom{k}{r}\int_{s=0}^{1}(s+\alpha)^r(-s-1)^{k-r}ds \\ &\quad + \sum_{r=d+1}^{k}\binom{k}{r}\int_{s=0}^{1}(s+\alpha)^r(-s-1)^{k-r}ds \\ &= \int_{s=0}^{1}\sum_{r=0}^{k}\binom{k}{r}(s+\alpha)^r(-s-1)^{k-r}ds \\ &= (\alpha-1)^k.\end{aligned}$$

ce qui complète la preuve. □

3.3.2 Erreur numérique

L'erreur numérique au point x_{ij} et au temps t^n est définie par

$$e_{ij}^n = f(t^n, x_{ij}) - \mathcal{S}^n(f(t^0, \mathbf{x}))_{ij}$$

Nous relions classiquement l'erreur numérique à l'erreur de troncature :

Proposition 3.3.6. *Nous avons*

$$\boldsymbol{e}^n = \mathcal{S}\boldsymbol{e}^{n-1} + \Delta t \boldsymbol{g}^{n-1}.$$

Par itération, nous obtenons

$$\boldsymbol{e}^n = \Delta t \sum_{\ell=0}^{n-1} \mathcal{S}^\ell \boldsymbol{g}^{n-1-\ell}. \tag{3.3.1}$$

Démonstration. Par définition, nous avons

$$\mathbf{e}^n = f(t^n, \mathbf{x}) - \mathcal{S}(\mathcal{S}^{n-1}(f(t^0, \mathbf{x})))$$

et

$$\mathcal{S}^{n-1}(f(t^0, \mathbf{x})) = f(t^{n-1}, \mathbf{x}) - \mathbf{e}^{n-1}.$$

ce qui conduit à

$$\mathbf{e}^n = \mathcal{S}\mathbf{e}^{n-1} + f(t^n, \mathbf{x}) - \mathcal{S}(f(t^{n-1}, \mathbf{x})),$$

d'où la conclusion. □

3.4 Analyse de la structure propre

Nous voyons dans l'expression de l'erreur numérique (3.3.1) que nous avons à considérer les puissances $\mathcal{S}$. Une telle étude peut être réalisée en regardant la décomposition spectrale de $\mathcal{S}$.

Pour chaque cellule $i = 0, \dots, N-1$, le schéma donné par (3.1.1) peut être écrit sous la forme

$$\begin{pmatrix} f_{i,0}^{n+1} \\ \vdots \\ f_{i,d}^{n+1} \end{pmatrix} = A_{-1} \begin{pmatrix} f_{i-1,0}^{n} \\ \vdots \\ f_{i-1,d}^{n} \end{pmatrix} + A_0 \begin{pmatrix} f_{i,0}^{n} \\ \vdots \\ f_{i,d}^{n} \end{pmatrix}$$

où $A_{-1}, A_0 \in \mathcal{M}_{d+1}(\mathbb{R})$ sont les matrices :

$$(A_{-1})_{ij} = \int_{\mathbb{R}} \varphi_j(\alpha + s)\varphi_i(s)ds = \int_{s=\alpha}^{1} \varphi_i(s-\alpha)\varphi_j(s)ds$$
$$(A_0)_{ij} = \int_{\mathbb{R}} \varphi_j(\alpha + s - 1)\varphi_i(s)ds = \int_{s=0}^{\alpha} \varphi_i(s-\alpha+1)\varphi_j(s)ds.$$

Ainsi, dans la base naturelle associée $\mathbf{x}$, la matrice de $\mathcal{S}$ est donnée par

$$\mathcal{S} = \begin{pmatrix} A_0 & & & A_{-1} \\ A_{-1} & A_0 & & \\ & \ddots & \ddots & \\ & & A_{-1} & A_0 \end{pmatrix} \in \mathcal{M}_{N(d+1)}(\mathbb{R})$$

La matrice $\mathcal{S}$ est une matrice circulante par blocs, nous pouvons donc effectuer une décomposition de Fourier vectorielle. Une telle décomposition est déjà utilisée dans [25].

Proposition 3.4.1. *Nous avons la décomposition*

$$\mathcal{S} = UDU^\star$$

où

$$D = \begin{pmatrix} D_0 & & \\ & \ddots & \\ & & D_{N-1} \end{pmatrix} \qquad U = \begin{pmatrix} U_{0,0} & \dots & U_{0,N-1} \\ \vdots & \ddots & \vdots \\ U_{N-1,0} & \dots & U_{N-1,N-1} \end{pmatrix}$$

avec

$$D_m = A_0 + A_{-1}e^{\frac{2i\pi m}{N}} \qquad U_{k,\ell} = \frac{1}{\sqrt{N}} e^{\frac{2i\pi k\ell}{N}} I_{d+1}$$

et I_{d+1} est la matrice identité de taille $(d+1)\times(d+1)$.

Démonstration. Le (k,ℓ)-bloc de la matrice $UDU^\star$ est égal à

$$\frac{1}{N}\sum_{m=0}^{N-1} D_m e^{\frac{2i\pi m(k-\ell)}{N}} = A_0\delta_{k\ell} + A_{-1}\delta_{k+1,\ell}$$

où δ est le symbole de Kronecker. □

Nous cherchons maintenant la structure propre de $\mathcal{S}$. Pour cela, nous pouvons nous réduire à étudier la structure propre des matrices $D_k, k = 0, \ldots, N-1$. Nous considérons d'abord la matrice D_0 :

Proposition 3.4.2. *La seule valeur propre de module 1 de la matrice $D_0 := A_{-1} + A_0$ est 1, qui n'est pas multiple. Les autres valeurs propres ont un module strictement inférieur à 1. De plus, nous avons*

$$D_0^n \underset{n\to+\infty}{\longrightarrow} G$$

où G désigne la matrice des poids de Gauss :

$$G = \begin{pmatrix} \omega_0 & \omega_1 & \ldots & \omega_d \\ \omega_0 & \omega_1 & \ldots & \omega_d \\ \vdots & \vdots & \vdots & \vdots \\ \omega_0 & \omega_1 & \ldots & \omega_d \end{pmatrix}.$$

Démonstration. Dans cette preuve, nous utiliserons les relations suivantes :

$$\sum_{j=0}^{d} \varphi_j(x) = 1 \qquad \text{pour tout } 0 \leq x \leq 1 \tag{3.4.1}$$

$$\int_0^1 \varphi_j(s)ds = \omega_j. \tag{3.4.2}$$

Soit λ une valeur propre de la matrice D_0. Il existe $x = (x_0, ..., x_d)$ tel que $D_0 x = \lambda x$:

$$\lambda\omega_j x_j = \sum_{j'=0}^{d} x_{j'} \int_{s=\alpha}^{1} \varphi_{j'}(s)\varphi_j(s-\alpha)ds + \sum_{j'=0}^{d} x_{j'} \int_{s=0}^{\alpha} \varphi_{j'}(s)\varphi_j(s+1-\alpha)ds.$$

Nous notons par P le polynôme de degré inférieur ou égal à d tel que $P(\alpha_j) = x_j$ pour tout $j = 0, ..., d$. Nous obtenons donc

$$\begin{aligned} \lambda\omega_j P(\alpha_j) &= \sum_{j'=0}^{d} P(\alpha_{j'}) \int_{s=\alpha}^{1} \varphi_{j'}(s)\varphi_j(s-\alpha)ds \\ &\quad + \sum_{j'=0}^{d} P(\alpha_{j'}) \int_{s=0}^{\alpha} \varphi_{j'}(s)\varphi_j(s+1-\alpha)ds. \end{aligned} \tag{3.4.3}$$

Nous sommons sur j :

$$\begin{aligned} \lambda \sum_{j=0}^{d} \omega_j P(\alpha_j) &= \sum_{j'=0}^{d} P(\alpha_{j'}) \int_{s=\alpha}^{1} \varphi_{j'}(s) \sum_{j=0}^{d} \varphi_j(s-\alpha)ds \\ &\quad + \sum_{j'=0}^{d} P(\alpha_{j'}) \int_{s=0}^{\alpha} \varphi_{j'}(s) \sum_{j=0}^{d} \varphi_j(s+1-\alpha)ds \end{aligned}$$

et nous obtenons, par (3.4.1) et par la formule de quadrature de Gauss,

$$\lambda \int_{s=0}^{1} P(s)ds = \sum_{j'=0}^{d} P(\alpha_{j'}) \int_{s=\alpha}^{1} \varphi_{j'}(s)ds + \sum_{j'=0}^{d} P(\alpha_{j'}) \int_{s=0}^{\alpha} \varphi_{j'}(s)ds$$

puis nous utilisons (3.4.2) ainsi que la formule de quadrature de Gauss pour obtenir finalement

$$\lambda \int_{s=0}^{1} P(s)ds = \int_{s=0}^{1} P(s)ds. \tag{3.4.4}$$

En conclusion, si $\lambda \neq 1$ alors $\int_{s=0}^{1} P(s)ds = 0$.

En multipliant (3.4.3) par $P(\alpha_j)$:

$$\begin{aligned} \lambda\omega_j P^2(\alpha_j) &= \sum_{j'=0}^{d} P(\alpha_{j'})P(\alpha_j) \int_{s=\alpha}^{1} \varphi_{j'}(s)\varphi_j(s-\alpha)ds \\ &\quad + \sum_{j'=0}^{d} P(\alpha_{j'})P(\alpha_j) \int_{s=0}^{\alpha} \varphi_{j'}(s)\varphi_j(s+1-\alpha)ds. \end{aligned}$$

Nous sommons sur j :

$$\begin{aligned} \lambda \sum_{j=0}^{d} \omega_j P^2(\alpha_j) &= \int_{s=\alpha}^{1} \left(\sum_{j'=0}^{d} P(\alpha_{j'})\varphi_{j'}(s) \right) \left(\sum_{j=0}^{d} P(\alpha_j)\varphi_j(s-\alpha) \right) ds \\ &\quad + \int_{s=0}^{\alpha} \left(\sum_{j'=0}^{d} P(\alpha_{j'})\varphi_{j'}(s) \right) \left(\sum_{j=0}^{d} P(\alpha_j)\varphi_j(s+1-\alpha) \right) ds \end{aligned}$$

et nous obtenons, puisque $\deg(P^2) \leq 2d \leq 2d+1$ et la formule de quadrature de Gauss est toujours valable :

$$\lambda \int_{0}^{1} P(s)^2 ds = \int_{0}^{1} P(s)(1_{[\alpha,1[}(s)P(s-\alpha) + 1_{[0,\alpha]}(s)P(s+1-\alpha))ds. \tag{3.4.5}$$

Par l'inégalité de Cauchy-Schwarz, nous avons

$$\begin{aligned} &\left(\int_0^1 P(s)(1_{[\alpha,1[}(s)P(s-\alpha) + 1_{[0,\alpha]}(s)P(s+1-\alpha))ds \right)^2 \leq \\ &\int_0^1 P(s)^2 ds \cdot \int_0^1 (1_{[\alpha,1[}(s)P(s-\alpha) + 1_{[0,\alpha]}(s)P(s+1-\alpha))^2 ds. \end{aligned}$$

Le dernier terme peut être simplifié :

$$\begin{aligned} &\int_0^1 (1_{[\alpha,1[}(s)P(s-\alpha) + 1_{[0,\alpha]}(s)P(s+1-\alpha))^2 ds \\ &= \int_0^{1-\alpha} P(s)^2 ds + \int_{1-\alpha}^1 P(s)^2 ds = \int_0^1 P(s)^2 ds \end{aligned}$$

ainsi nous obtenons :

$$\left|\int_0^1 P(s)(1_{[\alpha,1[}(s)P(s-\alpha)+1_{[0,\alpha]}(s)P(s+1-\alpha))ds\right| \leq \int_0^1 P(s)^2 ds. \tag{3.4.6}$$

Les relations (3.4.5) et (3.4.6) conduisent à

$$|\lambda| \int_0^1 P(s)^2 ds \leq \int_0^1 P(s)^2 ds. \tag{3.4.7}$$

Il y a égalité dans (3.4.6) si et seulement si les fonctions $s \mapsto P(s)$ et $s \mapsto 1_{[\alpha,1[}(s)P(s-\alpha)+1_{[0,\alpha]}(s)P(s+1-\alpha)$ sont proportionnelles *i.e.* il existe $(\mu_1,\mu_2) \neq (0,0)$ tel que

$$\mu_1 P(s) = \mu_2(1_{[\alpha,1[}(s)P(s-\alpha)+1_{[0,\alpha]}(s)P(s+1-\alpha)).$$

Il est clair qu'une telle relation est possible si P est de degré 1 (nous supposons que $0<\alpha<1$) et si P est de degré supérieur ou égal à 1, nous pouvons dériver la relation qui reste la même pour les dérivées qui seront de degré 1 à partir d'un certain moment. Ainsi P est nécessairement constant.

Si $|\lambda| = 1$ nous avons égalité dans (3.4.7) et donc, par la remarque précédente, P est constant. Ainsi, si $|\lambda| = 1$ et $\lambda \neq 1$, la relation (3.4.4) implique $P = 0$ ce qui n'est pas possible.

nous considérons les deux sous-espaces $\mathcal{V} = \{P \in \mathbb{C}_d[X] \mid P \text{ is constant}\}$ et $\mathcal{W} = \{P \in \mathbb{C}_d[X] \mid \int_0^1 P(x)dx = 0\}$. Ces deux sous-espaces sont en somme directe associés à la décomposition $P(X) = \int_0^1 P(x)dx + \left(P(X) - \int_0^1 P(x)dx\right)$. Ces sous-espaces sont de plus stables par D_0.

En effet, si $P \in \mathcal{V}$ nous pouvons supposer que $P \equiv 1$. La j^{eme}-composante de $D_0 \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$ s'écrit

$$\frac{1}{\omega_j}\sum_{j'=0}^{d}\int_{s=\alpha}^{1}\varphi_{j'}(s)\varphi_j(s-\alpha)ds + \frac{1}{\omega_j}\sum_{j'=0}^{d}\int_{s=0}^{\alpha}\varphi_{j'}(s)\varphi_j(s+1-\alpha)ds = \frac{1}{\omega_j}\int_0^1 \varphi_j(s)ds = 1.$$

Si $P \in \mathcal{W}$, nous avons $\int_0^1 P = \sum_{i=0}^d \omega_i P(\alpha_i) = 0$ alors $\int_0^1 D_0 P$ s'écrit

$$\begin{aligned}
& \sum_{j=0}^{d} \left(\sum_{j'=0}^{d} P(\alpha_{j'}) \int_{s=\alpha}^{1} \varphi_{j'}(s)\varphi_j(s-\alpha)ds + \sum_{j'=0}^{d} P(\alpha_{j'}) \int_{s=0}^{\alpha} \varphi_{j'}(s)\varphi_j(s+1-\alpha)ds \right) \\
= & \sum_{j'=0}^{d} P(\alpha_{j'}) \int_{s=\alpha}^{1} \varphi_{j'}(s) \sum_{j=0}^{d} \varphi_j(s-\alpha)ds + \sum_{j'=0}^{d} P(\alpha_{j'}) \int_{s=0}^{\alpha} \varphi_{j'}(s) \sum_{j=0}^{d} \varphi_j(s+1-\alpha)ds \\
= & \sum_{j'=0}^{d} P(\alpha_{j'}) \int_{s=0}^{1} \varphi_{j'}(s)ds \\
= & \sum_{j'=0}^{d} \omega_{j'} P(\alpha_{j'}) \\
= & 0.
\end{aligned}$$

En résumé, la matrice D_0 admet la valeur propre $\lambda = 1$ associée à l'espace propre $\mathcal{V}$ de dimension 1 et les autres valeurs propres sont de module strictement inférieur à 1 associé à l'espace $\mathcal{W}$ de dimension d.

Le vecteur de base e_i correspond au polynôme P_i défini par $P_i(\alpha_j) = \delta_{ij}$. La projection de P_i sur le sous-espace $\mathcal{V}$ donne $\int P_i = \sum_j P_i(\alpha_j)\omega_j = \omega_i$. Ainsi, nous avons

$$\lim_{n \to +\infty} D_0^n e_i = \omega_i \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$$

ce qui complète la preuve. □

Remarque 3.4.3. *Les valeurs propres de la matrice $D_0 := A_{-1} + A_0$ dépendent de $\alpha = \frac{a\Delta x}{\Delta t}$ qui est ici supposé constant. Ainsi, les valeurs propres ne dépendent pas du maillage et $\lambda^n \to 0$, quand n tend vers l'infini si $|\lambda| < 1$.*

Nous considérons alors les autres matrices $D_1, \ldots, D_{N-1}$:

Proposition 3.4.4. *Pour tout $m = 1 \ldots N-1$, la matrice $D_m = A_0 + A_{-1} e^{\frac{2i\pi m}{N}}$ vérifie*

$$D_m^n \underset{n \to +\infty}{\longrightarrow} 0$$

Démonstration. Dans cette preuve, nous utiliserons comme précédemment les relations suivantes :

$$\sum_{j=0}^{d} \varphi_j(x) = 1 \qquad \text{for all } 0 \le x \le 1 \tag{3.4.8}$$

$$\int_0^1 \varphi_j(s)ds = \omega_j. \tag{3.4.9}$$

Soit λ une valeur propre de la matrice D_m. Il existe $x = (x_0, ..., x_d)$ tel que $D_m x = \lambda x$:

$$\lambda\omega_j x_j = e^{\frac{2i\pi m}{N}} \sum_{j'=0}^{d} x_{j'} \int_{s=\alpha}^{1} \varphi_{j'}(s)\varphi_j(s-\alpha)ds + \sum_{j'=0}^{d} x_{j'} \int_{s=0}^{\alpha} \varphi_{j'}(s)\varphi_j(s+1-\alpha)ds.$$

Nous notons par P le polynôme de degré inférieur ou égal à d tel que $P(\alpha_j) = x_j$ pour tout $j = 0, ..., d$. Alors :

$$\begin{aligned} \lambda\omega_j P(\alpha_j) &= e^{\frac{2i\pi m}{N}} \sum_{j'=0}^{d} P(\alpha_{j'}) \int_{s=\alpha}^{1} \varphi_{j'}(s)\varphi_j(s-\alpha)ds \\ &\quad + \sum_{j'=0}^{d} P(\alpha_{j'}) \int_{s=0}^{\alpha} \varphi_{j'}(s)\varphi_j(s+1-\alpha)ds. \end{aligned} \tag{3.4.10}$$

Nous sommons sur j :

$$\begin{aligned} \lambda \sum_{j=0}^{d} \omega_j P(\alpha_j) &= e^{\frac{2i\pi m}{N}} \sum_{j'=0}^{d} P(\alpha_{j'}) \int_{s=\alpha}^{1} \varphi_{j'}(s) \sum_{j=0}^{d} \varphi_j(s-\alpha)ds \\ &\quad + \sum_{j'=0}^{d} P(\alpha_{j'}) \int_{s=0}^{\alpha} \varphi_{j'}(s) \sum_{j=0}^{d} \varphi_j(s+1-\alpha)ds \end{aligned}$$

et nous obtenons, par (3.4.8) et par la formule de quadrature de Gauss,

$$\lambda \int_{s=0}^{1} P(s)ds = e^{\frac{2i\pi m}{N}} \sum_{j'=0}^{d} P(\alpha_{j'}) \int_{s=\alpha}^{1} \varphi_{j'}(s)ds + \sum_{j'=0}^{d} P(\alpha_{j'}) \int_{s=0}^{\alpha} \varphi_{j'}(s)ds$$

Alors nous utilisons (3.4.9) et la formule de quadrature de Gauss pour finalement obtenir

$$(\lambda - 1) \int_{s=0}^{1} P(s)ds = (e^{\frac{2i\pi m}{N}} - 1) \int_{s=\alpha}^{1} P(s)ds. \tag{3.4.11}$$

Nous multiplions (3.4.10) par $P(\alpha_j)$:

$$\begin{aligned} \lambda\omega_j P^2(\alpha_j) &= e^{\frac{2i\pi m}{N}} \sum_{j'=0}^{d} P(\alpha_{j'})P(\alpha_j) \int_{s=\alpha}^{1} \varphi_{j'}(s)\varphi_j(s-\alpha)ds \\ &\quad + \sum_{j'=0}^{d} P(\alpha_{j'})P(\alpha_j) \int_{s=0}^{\alpha} \varphi_{j'}(s)\varphi_j(s+1-\alpha)ds. \end{aligned}$$

Nous sommons sur j :

$$\begin{aligned} \lambda \sum_{j=0}^{d} \omega_j P^2(\alpha_j) = e^{\frac{2i\pi m}{N}} &\int_{s=\alpha}^{1} \left(\sum_{j'=0}^{d} P(\alpha_{j'})\varphi_{j'}(s) \right) \left(\sum_{j=0}^{d} P(\alpha_j)\varphi_j(s-\alpha) \right) ds \\ &+ \int_{s=0}^{\alpha} \left(\sum_{j'=0}^{d} P(\alpha_{j'})\varphi_{j'}(s) \right) \left(\sum_{j=0}^{d} P(\alpha_j)\varphi_j(s+1-\alpha) \right) ds \end{aligned}$$

et nous obtenons, puisque $\deg(P^2) \leq 2d \leq 2d+1$ et que la formule de quadrature de Gauss est toujours valable :

$$\lambda \int_0^1 P(s)^2 ds = \int_0^1 P(s)(1_{[\alpha,1[}(s) e^{\frac{2i\pi m}{N}} P(s-\alpha) + 1_{[0,\alpha]}(s) P(s+1-\alpha)) ds. \tag{3.4.12}$$

Par l'inégalité de Cauchy-Schwarz, nous avons

$$\left| \int_0^1 P(s)(1_{[\alpha,1[}(s) e^{\frac{2i\pi m}{N}} P(s-\alpha) + 1_{[0,\alpha]}(s) P(s+1-\alpha)) ds \right|^2 \leq$$
$$\int_0^1 P(s)^2 ds \cdot \int_0^1 \left| 1_{[\alpha,1[}(s) e^{\frac{2i\pi m}{N}} P(s-\alpha) + 1_{[0,\alpha]}(s) P(s+1-\alpha) \right|^2 ds.$$

Le dernier terme peut être simplifié, puisque les fonctions sont à supports disjoints :

$$\int_0^1 \left| 1_{[\alpha,1[}(s) e^{\frac{2i\pi m}{N}} P(s-\alpha) + 1_{[0,\alpha]}(s) P(s+1-\alpha) \right|^2 ds$$
$$= \int_0^1 \left| 1_{[\alpha,1[}(s) e^{\frac{2i\pi m}{N}} P(s-\alpha) \right|^2 + \int_0^1 \left| 1_{[0,\alpha]}(s) P(s+1-\alpha) \right|^2 ds = \int_0^1 P(s)^2 ds$$

d'où :

$$\left| \int_0^1 P(s)(1_{[\alpha,1[}(s) e^{\frac{2i\pi m}{N}} P(s-\alpha) + 1_{[0,\alpha]}(s) P(s+1-\alpha)) ds \right| \leq \int_0^1 P(s)^2 ds \tag{3.4.13}$$

et les relations (3.4.12) et (3.4.13) conduisent à

$$|\lambda| \int_0^1 P(s)^2 ds \leq \int_0^1 P(s)^2 ds. \tag{3.4.14}$$

Nous avons l'égalité dans (3.4.13) si et seulement si les fonctions $s \mapsto P(s)$ et $s \mapsto 1_{[\alpha,1[}(s) e^{\frac{2i\pi m}{N}} P(s-\alpha) + 1_{[0,\alpha]}(s) P(s+1-\alpha)$ sont proportionnelles *i.e.* il existe $(\mu_1, \mu_2) \neq (0,0)$ tel que

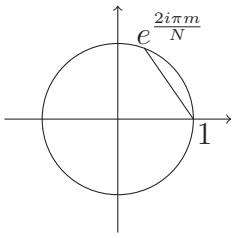
$$\mu_1 P(s) = \mu_2 (1_{[\alpha,1[}(s) e^{\frac{2i\pi m}{N}} P(s-\alpha) + 1_{[0,\alpha]}(s) P(s+1-\alpha)).$$

Il est clair qu'une telle relation n'est pas possible si P est de degré 1 (nous supposons $0 < \alpha < 1$) et si P est de degré supérieur à 1, alors nous pouvons dériver cette relation qui reste la même pour les dérivées successives qui seront de degré 1 à un moment. Ainsi P est nécessairement constant.

Si $|\lambda| = 1$ nous avons égalité dans (3.4.14) et donc, par la remarque précédente, P est constant. Dans ce cas, nous obtenons, en utilisant (3.4.11) :

$$\lambda = (1-\alpha) e^{\frac{2i\pi m}{N}} + \alpha$$

ainsi nous avons prouvé le fait que λ appartient au segment entre 1 et $e^{\frac{2i\pi m}{N}}$ et finalement la condition $0 < \alpha < 1$ implique que $|\lambda| < 1$.



Au final, toutes les valeurs propres ont un module strictement inférieur à 1, ce qui complète la preuve. □

En utilisant les deux propriétés précedentes, nous pouvons établir que :

Corollaire 3.4.5. *Nous avons la propriété de convergence :*

$$\mathcal{S}^n \underset{n\to+\infty}{\longrightarrow} UD^\infty U^\star$$

où

$$D^\infty = \begin{pmatrix} G & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \end{pmatrix}.$$

Remarque 3.4.6. *Les matrices $D_m = A_0 + A_1 e^{-\frac{2i\pi m}{N}}$ pour $m = 1, ..., N-1$ dépendent de N et leurs valeurs propres dépendent ainsi de Δx.*

3.5 Discussion autour de la preuve pour un degré quelconque

D'après la proposition 3.3.6, nous avons

$$\mathbf{e}^n = \Delta t \sum_{\ell=0}^{n-1} \mathcal{S}^\ell \mathbf{g}^{n-1-\ell},$$

ce qui conduit à

$$\|\mathbf{e}^n\|_2 \leq \Delta t \underbrace{\sum_{\ell=0}^{n-1} \|\mathcal{S}^\ell - UD^\infty U^\star\|_2 \|\mathbf{g}^{n-1-\ell}\|_2}_{(1)} + \Delta t \underbrace{\left\| \sum_{\ell=0}^{n-1} UD^\infty U^\star \mathbf{g}^{n-1-\ell} \right\|_2}_{(2)}.$$

Premier terme. Concernant le terme (1), nous faisons la conjecture suivante (non démontrée) :

$$\sum_{\ell=0}^{n-1} \|\mathcal{S}^\ell - UD^\infty U^\star\|_2 \|\mathbf{g}^{n-1-\ell}\|_2 \leq C_1 \frac{\Delta x^{d+1}}{\Delta t} + nC_2 \frac{\Delta x^{2d+2}}{\Delta t}$$

où $C_1, C_2 > 0$ sont des constantes dépendantes de α.

Second terme : nous pouvons montrer, par calcul, que le vecteur $UD^\infty U^\star \mathbf{g}^\ell$ est égal à

$$\left(\sum_{j=0}^{d} \omega_j \sum_{i=0}^{N-1} g_{ij}^\ell \right) \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \in \mathbb{R}^{(d+1)N}$$

et nous avons, d'après la proposition 3.3.3 :

$$\begin{aligned}\sum_{i=0}^{N-1}\sum_{j=0}^{d}\omega_j\sum_{\ell=0}^{n-1}g_{ij}^{\ell} &= \sum_{\ell=0}^{n-1}\sum_{k=0}^{2d+1}\sum_{j=0}^{d}\omega_j E_j^k\frac{\Delta x^k}{\Delta t}\left(\sum_{i=0}^{N-1}\partial_x^k f(t^\ell,x_{ij})\right)+\mathcal{O}\left(\frac{\Delta x^{2d+2}}{\Delta t}\right)\\ &= \sum_{\ell=0}^{n-1}\sum_{k=0}^{2d+1}\sum_{j=0}^{d}\omega_j E_j^k\frac{\Delta x^k}{\Delta t}\left(\sum_{i=0}^{N-1}\partial_x^k f^0(x_{ij}-\ell\Delta t)\right)+\mathcal{O}\left(\frac{\Delta x^{2d+2}}{\Delta t}\right).\end{aligned}$$

Le théorème d'Euler-MacLaurin établit que si $(m,n)\in\mathbb{Z}^2$, $m<n$, $k\in\mathbb{N}^*$ et $f:[m,n]\to\mathbb{C}$ une fonction $\mathcal{C}^r([m,n])$ alors nous avons :

$$\frac{f(m)}{2}+f(m+1)+\ldots+f(n-1)+\frac{f(n)}{2}=\int_m^n f(t)dt+\sum_{k=2}^{r}\frac{b_k}{k!}(f^{(k-1)}(n)-f^{(k-1)}(m))+R_r$$

avec

$$R_r=\frac{(-1)^{r+1}}{r!}\int_m^n\tilde{B}_r(t)f^{(r)}(t)dt$$

où b_n sont les nombres de Bernoulli et $\tilde{B}_n$ les polynômes de Bernoulli. Nous appliquons le théorème de Euler-MacLaurin à la fonction

$$i\mapsto\partial_x^k f^0(x_{ij}-\ell\Delta t)=\partial_x^k f^0((i+\alpha_j)\Delta x-\ell\Delta t)$$

avec $r=2d+2-k$:

$$\begin{aligned}\sum_{i=0}^{N-1}\partial_x^k f^0((i+\alpha_j)\Delta x-\ell\Delta t) &= \int_{t=0}^{N}\partial_x^k f^0((t+\alpha_j)\Delta x-\ell\Delta t)dt+R_{2d+2-k}^{j,\ell}\\ &= \frac{1}{\Delta x}\int_0^1\partial_x^k f^0(x)dx+R_{2d+2-k}^{j,\ell}\end{aligned}$$

où

$$R_{2d+2-k}^{j,\ell}=\frac{(-1)^{2d+3-k}}{(2d+2-k)!}\int_0^N\tilde{B}_{2d+2-k}(t)\partial_x^k f^{0^{(2d+2-k)}}((t+\alpha_j)\Delta x-\ell\Delta t)dt.$$

D'où, nous obtenons

$$\begin{aligned}\sum_{i=0}^{N-1}\sum_{j=0}^{d}\omega_j\sum_{\ell=0}^{n-1}g_{ij}^{\ell} &= \sum_{\ell=0}^{n-1}\sum_{k=0}^{2d+1}\frac{\Delta x^k}{\Delta t}\left(\frac{1}{\Delta x}\int_0^1\partial_x^k f^0(x)dx\right)\left(\sum_{j=0}^{d}\omega_j E_j^k\right)\\ &\quad+\sum_{\ell=0}^{n-1}\sum_{k=0}^{2d+1}\sum_{j=0}^{d}\omega_j E_j^k\frac{\Delta x^k}{\Delta t}R_{2d+2-k}^{j,\ell}+\mathcal{O}\left(\frac{\Delta x^{2d+2}}{\Delta t}\right)\end{aligned}$$

Nous utilisons la propriété 3.3.5 pour conclure à

$$\sum_{\ell=0}^{n-1}\sum_{k=0}^{2d+1}\frac{\Delta x^k}{\Delta t}\left(\frac{1}{\Delta x}\int_0^1\partial_x^k f^0(x)dx\right)\left(\sum_{j=0}^{d}\omega_j E_j^k\right)=0$$

et $R_{2d+2-k}^{j,\ell}=\mathcal{O}(\Delta x^{2d+2-k})$ conduit à

$$\left\|\sum_{\ell=0}^{n-1}UD^{\infty}U^{\star}\mathbf{g}^{n-1-\ell}\right\|_2\leq nC_2\frac{\Delta x^{2d+2}}{\Delta t}$$

ce qui conclut l'estimation du second terme.

3.6 Résultats formels et numériques

3.6.1 Résultats formels

Le schéma est donné par $F_j^{n+1} = A_{-1}F_{j-1}^n + A_0F_j^n$ (voir le début de la partie 3.4). Nous rappelons que $\omega := 2\pi k\Delta x$ et

$$\widehat{F}_\omega^n = \frac{1}{N}\sum_{\ell=0}^{N-1} F_\ell^n \exp(-\mathrm{i}\ell\omega).$$

En considérant la condition initiale :

$$f(0,x) = \exp(2\mathrm{i}\pi kx)$$

nous obtenons :

$$(\hat{F}_\omega^0)_j = \exp(\mathrm{i}\omega\alpha_j).$$

La matrice d'amplification du schéma $\widehat{F}_\omega^{n+1} = D_\omega\widehat{F}_\omega^n$ s'écrit

$$D_\omega = A_0 + A_{-1}\exp(-i\omega).$$

Nous notons par $\lambda_{0,\omega},...,\lambda_{d,\omega}$ et $V_{0,\omega},...,V_{d,\omega}$ les valeurs propres et vecteurs propres de D_ω. La soltuion du schéma est donnée par :

$$\widehat{F}_\omega^n = (D_\omega)^n\,\widehat{F}_\omega^0.$$

Nous choisissons les vecteurs propres $V_{0,\omega},...,V_{d,\omega}$ tels que $\widehat{F}_\omega^0 = \sum_{j=0}^d V_{j,\omega}(\alpha)$ alors nous avons

$$\widehat{F}_\omega^n = \sum_{j=0}^d \lambda_{j,\omega}^n V_{j,\omega}(\alpha).$$

L'erreur dans l'espace de Fourier s'écrit :

$$\sum_{j=0}^d (\lambda_{j,\omega}^n - e^{-i\omega n\alpha})V_{j,\omega}(\alpha).$$

Cas $d = 1$.

Nous obtenons les valeurs propres et vecteurs propres en utilisant Maple :

$$\begin{aligned}
\lambda_{0,\omega} &= 1 - \alpha i\omega + \frac{1}{2}(\alpha i\omega)^2 - \frac{1}{6}(\alpha i\omega)^3 + \frac{\alpha(4\alpha^3 - 2\alpha^2 + 2\alpha - 1)}{72}(i\omega)^4 + \mathcal{O}(\omega^5),\\
\lambda_{1,\omega} &= 6\alpha^2 - 6\alpha + 1 + \mathcal{O}(\omega),\\
V_{0,\omega} &= \begin{pmatrix} 1 + \frac{3-\sqrt{3}}{6}i\omega + \mathcal{O}(\omega^2) \\ 1 + \frac{3+\sqrt{3}}{6}i\omega + \mathcal{O}(\omega^2) \end{pmatrix},\\
V_{1,\omega} &= \begin{pmatrix} -\frac{\sqrt{3}(2\alpha-1)}{36}(i\omega)^2 - \frac{-4\sqrt{3}\alpha^2+(10\sqrt{3}-6)\alpha+(3-5\sqrt{3})}{216}(i\omega)^3 + \mathcal{O}(\omega^4) \\ \frac{\sqrt{3}(2\alpha-1)}{36}(i\omega)^2 - \frac{4\sqrt{3}\alpha^2+(-10\sqrt{3}-6)\alpha+(3+5\sqrt{3})}{216}(i\omega)^3 + \mathcal{O}(\omega^4) \end{pmatrix}.
\end{aligned}$$

Remarque 3.6.1. *Nous avons*

$$\lambda_{0,\omega} = e^{-i\alpha\omega} + \frac{\alpha(\alpha-1)(\alpha^2-\alpha+1)(i\omega)^4}{72} + \mathcal{O}(\omega^5)$$

avec $\alpha(\alpha-1)(\alpha^2-\alpha+1) < 0$ *pour* $0 < \alpha < 1$. *Ainsi, pour* $\omega \neq 0$ *et de petites valeurs de* Δx, *nous avons* $|\lambda_{0,\omega}| < 1$ *comme indiqué par la Proposition 3.4.4.*

L'erreur peut être décomposée comme suit :

$$\begin{aligned}\left\|\sum_{j=0}^{d}(\lambda_{j,\omega}^n - e^{-i\omega n\alpha})V_{j,\omega}(\alpha)\right\|_2 &\leq |\lambda_{0,\omega}^n - e^{-i\omega n\alpha}|\|V_{0,\omega}(\alpha)\|_2 \\ &\quad + |\lambda_{1,\omega}^n|\|V_{1,\omega}(\alpha)\|_2 + \|V_{1,\omega}(\alpha)\|_2.\end{aligned}$$

De plus, nous avons les majorations suivantes (où C désigne une constante dépendant uniquement de α et k) :

$$\begin{aligned}|\lambda_{0,k} - e^{-i\omega\alpha}| &\leq C\Delta x^4 \\ |\lambda_{0,k}^n - e^{-i\omega n\alpha}| &\leq nC\Delta x^4 \\ |\lambda_{1,\omega}| &\leq 1 \\ \|V_{0,\omega}(\alpha)\|_2 &\leq C \\ \|V_{1,\omega}(\alpha)\|_2 &\leq C\Delta x^2\end{aligned}$$

Nous validons, dans ce cas, l'estimation de l'erreur :

$$\|\mathbf{e}^n\|_2 \leq C_1\Delta x^{d+1} + nC_2\Delta x^{2d+2}.$$

Cas $d = 2$.

En raison de difficultés de calcul, nous ne pouvons pas obtenir les valeurs propres et vecteurs propres pour tout α. Ainsi, nous choisissons une valeur arbitraire de $\alpha = 0.27$ et nous calculons les valeurs propres et vecteurs propres pour cette valeur particulière. Nous obtenons :

$$\begin{aligned}\lambda_{0,\omega} &= 1 - \alpha i\omega + \frac{1}{2}(\alpha i\omega)^2 - \frac{1}{3!}(\alpha i\omega)^3 + \frac{1}{4!}(\alpha i\omega)^4 - \frac{1}{5!}(\alpha i\omega)^5 + C_1(\alpha)\omega^6 + \mathcal{O}(\omega^7), \\ \lambda_{1,\omega} &= C_2(\alpha) + \mathcal{O}(\omega), \\ \lambda_{2,\omega} &= C_3(\alpha) + \mathcal{O}(\omega)\end{aligned}$$

avec (pour $\alpha = 0.27$) :

$$C_1(\alpha = 0.27) \approx -1.42 \times 10^{-5} \neq \frac{1}{6!}(-\alpha\mathrm{i})^6 \approx -5.38 \times 10^{-7}$$

$$C_2(\alpha = 0.27) \approx -0.174 + 0.702\mathrm{i}, \quad C_3(\alpha = 0.27) \approx -0.174 - 0.702\mathrm{i}$$

$$
\begin{aligned}
V_{0,\omega} &= \begin{pmatrix} 1+\mathcal{O}(\omega) \\ 1+\mathcal{O}(\omega) \\ 1+\mathcal{O}(\omega) \end{pmatrix}, \\
V_{1,\omega} &= \begin{pmatrix} C_4(\alpha)(i\omega)^3+\mathcal{O}(\omega^4) \\ C_5(\alpha)(i\omega)^3+\mathcal{O}(\omega^4) \\ C_6(\alpha)(i\omega)^3+\mathcal{O}(\omega^4) \end{pmatrix}, \\
V_{2,\omega} &= \begin{pmatrix} C_7(\alpha)(i\omega)^3+\mathcal{O}(\omega^4) \\ C_8(\alpha)(i\omega)^3+\mathcal{O}(\omega^4) \\ C_9(\alpha)(i\omega)^3+\mathcal{O}(\omega^4) \end{pmatrix}
\end{aligned}
$$

avec (pour $\alpha = 0.27$) :

$$
\begin{aligned}
C_4(\alpha=0.27) \approx (5.56+10.4\mathrm{i})\times 10^{-4}, \quad C_5(\alpha=0.27) \approx (3.81-7.23\mathrm{i})\times 10^{-4}, \\
C_6(\alpha=0.27) \approx (-11.6+1.09\mathrm{i})\times 10^{-4}, \quad C_7(\alpha=0.27) \approx (-5.56+10.4\mathrm{i})\times 10^{-4}, \\
C_8(\alpha=0.27) \approx (-3.81-7.23\mathrm{i})\times 10^{-4}, \quad C_9(\alpha=0.27) \approx (11.6+1.09\mathrm{i})\times 10^{-4}
\end{aligned}
$$

Comme précédemment, nous avons les majorations :

$$
\begin{aligned}
|\lambda_{0,k} - e^{-i\omega\alpha}| &\leq C\Delta x^6 \\
|\lambda_{0,k}^n - e^{-i\omega n\alpha}| &\leq nC\Delta x^6 \\
|\lambda_{1,\omega}| &\leq 1 \\
\|V_{0,\omega}(\alpha)\|_2 &\leq C \\
\|V_{1,\omega}(\alpha)\|_2 &\leq C\Delta x^3
\end{aligned}
$$

et nous validons dans ce cas l'estimation de l'erreur :

$$\|\mathbf{e}^n\|_2 \leq C_1\Delta x^{d+1} + nC_2\Delta x^{2d+2}.$$

3.6.2 Résultats numériques

Nous procédons à une étude numérique de la convergence du schéma pour $d = 1$ et $d = 2$ (Fig. 3.1). Nous voyons que dans les deux cas, l'erreur est d'ordre $d+1$ lorsque nous itérons le schéma qu'une seule fois. En effet, pour un petit nombre d'itérations, le terme dominant de la borne de l'erreur

$$\|\mathbf{e}^n\|_2 \leq C_1\Delta x^{d+1} + nC_2\Delta x^{2d+2}$$

est $C_1\Delta x^{d+1}$. Lorsque le nombre d'itérations n augmente, le terme dominant de l'erreur devient $nC_2\Delta x^{2d+2}$ et alors nous observons l'émergence d'une pente $2d+2$ pour les grandes valeurs de Δx.

3.7 Conclusion

Nous avons montré numériquement et formellement une propriété de superconvergence pour le schéma Galerkin discontinu semi-Lagrangien pour de petits degrés. L'établissement de la preuve pour un degré quelconque et l'adaptation de cette preuve lorsque α tend vers 0 dans le but de montrer la propriété pour le schéma étudié dans [26] pourra faire l'objet d'un futur travail. Une telle propriété de superconvergence dans le cas de l'équation de Vlasov-Poisson est un problème totalement ouvert.

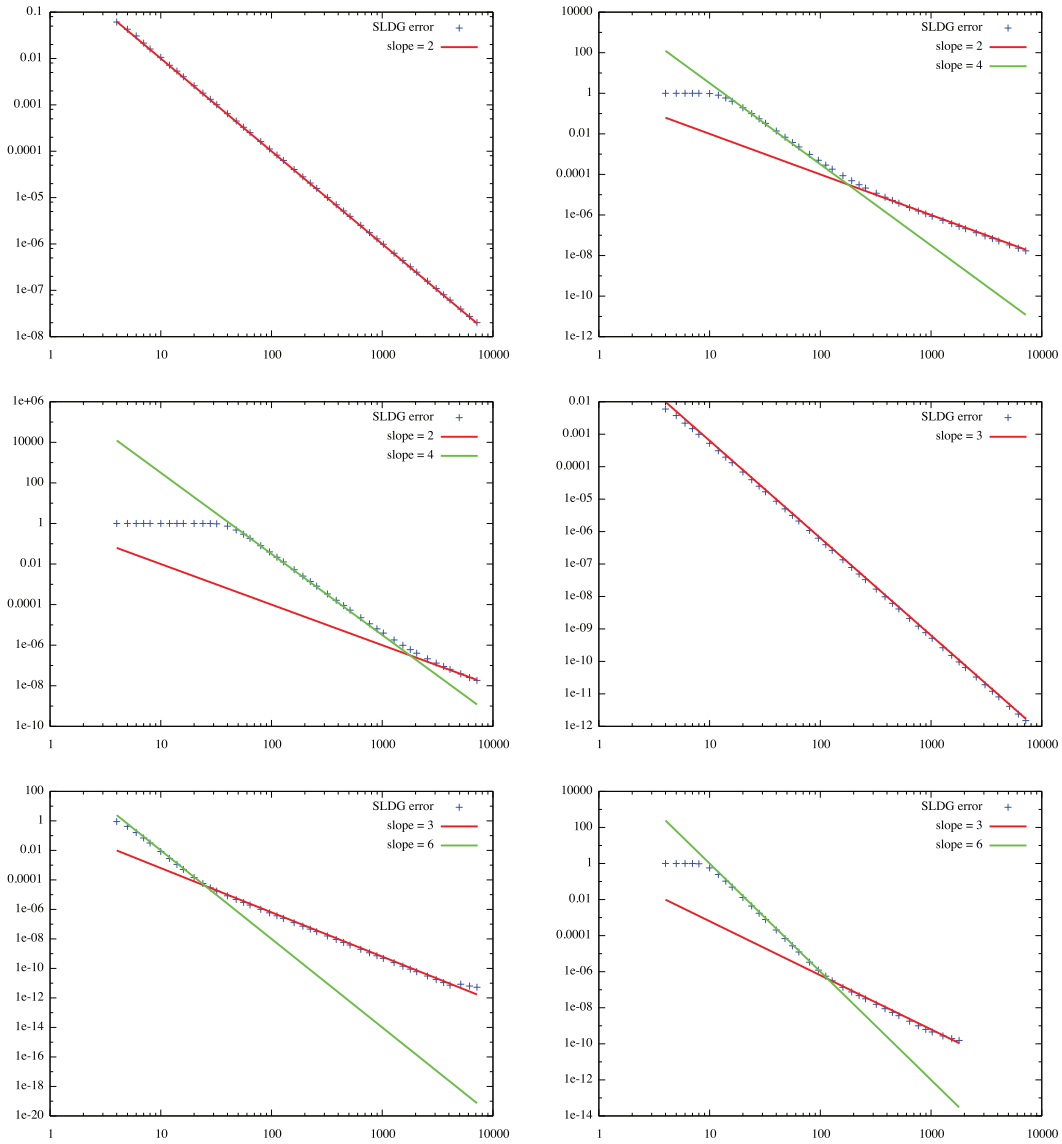


FIGURE 3.1 – Convergence de l'erreur pour le schéma SLGD pour $d = 1$ et 1 itération (en haut à gauche), 10^4 itérations (en haut à droite) et 10^6 itérations (au milieu à gauche). Erreur du schéma SLGD pour $d = 2$ et 1 itération (au milieu à droite), 10^4 itérations (en bas à gauche) et 10^6 itérations (en bas à droite). Nous avons utilisé $f_0(x) = \sin(2\pi x)$ et $a = 1$.

Deuxième partie

Méthodes pour l'équation de Vlasov-Poisson

La théorie cinétique des particules chargées interagissant avec un champ électrostatique et en ignorant les collisions peut être décrit par le système d'équations de Vlasov-Poisson. Ce modèle prend en compte l'évolution dans l'espace des phases d'une fonction de distribution $f(t, x, v)$ où $t \geq 0$ désigne le temps, x désigne l'espace et v est la vitesse.

En considérant le système unidimensionnel, on aboutit au modèle $1D \times 1D$ de Vlasov-Poisson où la solution $f(t, x, v)$ dépend du temps $t \geq 0$, de l'espace $x \in [0, L]$ et de la vitesse $v \in \mathbb{R}$. La fonction de distribution f satisfait

$$\partial_t f + v\partial_x f + E\partial_v f = 0, \tag{3.7.1}$$

où $E(t, x)$ est le champ électrique. La loi de Poisson impose que la distribution des particules chargées doit être sommée en vitesse pour que le champ électrique auto-consistant soit une solution de l'équation de Poisson :

$$\partial_x E = \int_{\mathbb{R}} f dv - 1. \tag{3.7.2}$$

Pour assurer l'unicité de la solution, nous imposons au champ électrique une condition de valeur moyenne nulle : $\int_0^L E(t, x)dx = 0$. Le système de Vlasov-Poisson (3.7.1)-(3.7.2) requiert une condition initiale $f(t = 0, x, v) = f_0(x, v)$. Nous attirons votre attention sur les conditions périodiques en espace et sur l'annulation de f pour les grandes vitesses.

A cause de la non-linéarité de l'évolution auto-consistante de deux champs interagissants, il est en général difficile de trouver une solution analytique à (3.7.1)-(3.7.2). Cela nécessite l'implémentation de méthodes numériques pour résoudre ce système. Historiquement, des progrès ont été faits en utilisant des méthodes particulaires (voir [53]) qui consistent à avancer en temps des macro-particules à travers les équations du mouvement tandis que le champ électrique est calculé sur un maillage spatial. Malgré le bruit numérique statistique inhérent et leur faible convergence, le coût de calcul des méthodes particulaires est très faible même en dimension élevée, ce qui explique leur forte popularité.

De l'autre côté se trouvent les méthodes Eulériennes qui ont été développées plus récemment et qui utilisent directement la grille de l'espace des phases (x, v). Les méthodes Eulériennes incluent les différences finies, les volumes finis ou les éléments finis. Clairement, ces méthodes sont très gourmandes en terme de mémoire mais peuvent converger très rapidement en utilisant des opérateurs discrets d'ordre élevé.

Entre ces deux familles de méthodes, les méthodes semi-Lagrangiennes essayent de garder les meilleures caractéristiques des deux approches : la fonction de distribution dans l'espace des phases est mise à jour en résolvant les équations du mouvement en arrière (*i.e.* les caractéristiques), et en utilisant une étape d'interpolation pour reconstruire la solution dans la grille de l'espace des phases. Ces méthodes sont souvent implémentées dans le contexte d'opérateur de splitting. Typiquement, pour résoudre (3.7.1)-(3.7.2), la stratégie consiste à décomposer le problème multidimensionnel en une suite de problèmes $1D$. Voir [35, 1, 42, 43, 31, 87, 47, 60] pour des travaux précédents sur le sujet.

Nous nous intéresserons plus particulièrement au cas test des ondes KEEN [51, 68] qui sont un cas test difficile pour les solveurs numériques de Vlasov-Poisson car ils nécessitent une haute résolution dans la région de l'espace des phases autour de la vitesse de guidage.

Chapitre 4

Méthodes de volumes finis pour Vlasov

Le contexte de l'équation de Vlasov-Poisson permet une opération de splitting licite. Cependant, dans certaines situations, cette procédure n'est pas appropriée [45, 76] et peut conduire à des instabilités numériques. L'objectif principal de ce chapitre est de rechercher des versions non splittées de schémas de volumes finis. De tels schémas ont déjà été développés dans [41] et, plus récemment, dans [33].

Nous détaillerons ici deux stratégies. La première suit [33] et conduit à un système d'EDO, avec upwind ou des approximations spatiales centrées. La deuxième stratégie consiste à approcher le flux avec des points de Gauss qui sont évalués en résolvant en arrière les caractéristiques ; ceci permet d'éviter le transport de volumes 2D, qui conduit à des calculs d'intersection de maillages [46, 50]. D'autres stratégies dans l'esprit de [41] peuvent être développées, mais ne le seront pas ici. Nous renvoyons également à [76] pour un travail récent dans cette direction.

Notre approche ici consiste à d'abord considérer l'advection linéaire unidimensionnelle (comme dans la méthode de splitting) afin d'analyser les propriétés de stabilité de ces deux types de schémas numériques dans un cadre simplifié. Cette information peut être un bon référent pour le contexte 2D. Nous faisons également un lien entre les approximations de type volumes finis et les schémas semi-Lagrangiens. En effet, lorsqu'une reconstruction de Lagrange est utilisée dans un schéma semi-Lagrangien, nous montrons que lorsque le pas de temps tend vers zéro, nous pouvons récupérer des approximations standards (upwind) des flux quand une approche volumes finis est utilisée. D'autres liens peuvent également être effectués.

4.1 Méthode des volumes finis de Banks

Cette section est consacrée à la présentation et à l'analyse d'une méthode de volumes finis [33]. Le cas 1D sera abordée conjointement avec une analyse de stabilité. Ensuite, nous détaillerons le cas 2D.

4.1.1 L'advection linéaire 1D

Nous considérons d'abord la résolution du problème d'advection linéaire 1D :

$$\begin{cases} \dfrac{\partial f(t,x)}{\partial t} + a\dfrac{\partial f(t,x)}{\partial x} = 0, \\ \\ f(t=0,x) = f^0(x), \end{cases} \tag{4.1.1}$$

où $f : [0,+\infty[\times \Omega \longrightarrow \mathbb{R}$ et $a \in \mathbb{R}$. Les inconnues sont les valeurs moyennes sur une cellule $\bar{f}_i^n = \frac{1}{\Delta x}\int_{x_{i-1/2}}^{x_{i+1/2}} f(t_n,x)dx$ et nous notons :

$$\bar{f}_i(t) = \frac{1}{\Delta x}\int_{x_{i-1/2}}^{x_{i+1/2}} f(t,x)dx.$$

En intégrant (4.1.1) sur un volume de contrôle et en divisant par sa taille Δx, nous obtenons :

$$\frac{d\bar{f}_i(t)}{dt} = -\frac{1}{\Delta x}\int_{x_{i-1/2}}^{x_{i+1/2}} a\partial_x f dx = -\frac{a}{\Delta x}[f(t,x_{i+1/2}) - f(t,x_{i-1/2})], \tag{4.1.2}$$

L'objectif est de donner un sens aux flux $f(t,x_{i\pm1/2})$ pour une suite $(\bar{f}_i(t))_i$ donnée.
Pour cela, nous écrivons classiquement

$$f(t,x_{i+1/2}) \approx \sum_{j=r}^{s} a_j \bar{f}_{i+j}(t), \tag{4.1.3}$$

Les conditions d'ordre nous permettent de déterminer les coefficients en résolvant le système

$$\frac{1}{2^k} = \sum_{j=r}^{s} a_j \int_{j-1/2}^{j+1/2} x^k dx, \;\; k = 0,\dots,r-s.$$

Notons que nous considérons ici des conditions aux bords périodiques.
Les discrétisations suivantes en espace vont être testées :

$$\begin{aligned}
CD2 : \quad & f(t,x_{i+1/2}) &\approx\;& \frac{1}{2}(\bar{f}_i(t) + \bar{f}_{i+1}(t)), \\
CD4 : \quad & f(t,x_{i+1/2}) &\approx\;& \frac{7}{12}(\bar{f}_i(t) + \bar{f}_{i+1}(t)) - \frac{1}{12}(\bar{f}_{i-1}(t) + \bar{f}_{i+2}(t)), \\
CD6 : \quad & f(t,x_{i+1/2}) &\approx\;& \frac{37}{60}(\bar{f}_i(t) + \bar{f}_{i+1}(t)) - \frac{8}{60}(\bar{f}_{i-1}(t) + \bar{f}_{i+2}(t)) + \frac{1}{60}(\bar{f}_{i-2}(t) + \bar{f}_{i+3}(t)), \\
UP1\ (a<0) : \quad & f(t,x_{i+1/2}) &\approx\;& \bar{f}_{i+1}(t), \\
UP3\ (a<0) : \quad & f(t,x_{i+1/2}) &\approx\;& \frac{1}{3}\bar{f}_i(t) + \frac{5}{6}\bar{f}_{i+1}(t) - \frac{1}{6}\bar{f}_{i+2}(t), \\
UP5\ (a<0) : \quad & f(t,x_{i+1/2}) &\approx\;& -\frac{1}{20}\bar{f}_{i-1}(t) + \frac{9}{20}\bar{f}_i(t) + \frac{47}{60}\bar{f}_{i+1}(t) - \frac{13}{60}\bar{f}_{i+2}(t) + \frac{1}{30}\bar{f}_{i+3}(t), \\
UP1\ (a>0) : \quad & f(t,x_{i+1/2}) &\approx\;& \bar{f}_i(t), \\
UP3\ (a>0) : \quad & f(t,x_{i+1/2}) &\approx\;& -\frac{1}{6}\bar{f}_{i-1}(t) + \frac{5}{6}\bar{f}_i(t) + \frac{1}{3}\bar{f}_{i+1}(t), \\
UP5\ (a>0) : \quad & f(t,x_{i+1/2}) &\approx\;& \frac{1}{30}\bar{f}_{i-2}(t) - \frac{13}{60}\bar{f}_{i-1}(t) + \frac{47}{60}\bar{f}_i(t) + \frac{9}{20}\bar{f}_{i+1}(t) - \frac{1}{20}\bar{f}_{i+2}(t).
\end{aligned}$$

	up 1	CD 2	up 3	CD 4	up 5	CD 6
RK 1	1.00	0.00	0.00	0.00	0.00	0.00
RK 2	1.00	0.00	0.87	0.00	0.00	0.00
RK 3	1.25	1.73	1.62	1.26	1.43	1.09
RK 4	1.39	2.82	1.74	2.06	1.73	1.78

TABLE 4.1 – Conditions CFL pour les schémas de volumes finis.

Une discrétisation temporelle classique avec un algorithme de Runge-Kutta explicite est ensuite utilisée et conduit au calcul de

$$\bar{f}^n = (\bar{f}^n_0, \dots, \bar{f}^n_{N-1}),\ f^n_j \simeq f_j(t_n),\ j = 0, \dots, N-1.$$

Plus précisément, l'approximation numérique de $y_n \approx y(t_n)$ du système différentiel obtenu par (4.1.2) écrit sous la forme $y'(t) = \phi(y(t))$ est donnée par

$$y_{n+1} \approx y_n + \Delta t \sum_{j=1}^{s} b_j k_j,\ k_j = \phi(y_n + \Delta t \sum_{\ell=1}^{j-1} a_{j,\ell} k_\ell),\ j = 1, \dots, s,$$

et nous avons considéré les exemples suivants :

$$\begin{aligned} &RK1\ (s=1) && b_1 = 1, \\ &RK2\ (s=2) && a_{2,1} = 1/2,\ b_1 = 0,\ b_2 = 1, \\ &RK3\ (s=3) && a_{2,1} = 1/2,\ a_{3,1} = -1,\ a_{3,2} = 2,\ b_1 = 1/6,\ b_2 = 2/3,\ b_3 = 1/6. \end{aligned}$$

et le schéma classique RK4 :

$$\begin{aligned} RK4\ (s=4) \qquad & a_{2,1} = 1/2,\ a_{3,1} = 0,\ a_{3,2} = 1/2,\ a_{4,1} = a_{4,2} = 0, \\ & a_{4,3} = 1,\ b_1 = b_4 = 1/6,\ b_2 = b_3 = 1/3. \end{aligned}$$

4.1.2 Stabilité et ordre

Nous avons déterminé une limite supérieure au-dessus de laquelle les schémas sont instables, ce qui revient à déterminer la condition CFL de ces schémas. Nous pouvons voir dans la Table 4.1 les CFL que nous avons trouvées (voir aussi [32]).
Un exemple d'un tel calcul est donné dans le cas CD4 RK1 (Euler). Nous effectuons une analyse de stabilité de Von Neumann. Pour cela, nous introduisons

$$f^n_j = \sum_{k=0}^{N-1} \hat{f}^n_k \exp(ikj\Delta x), \quad \text{avec} \quad \hat{f}^n_j = \frac{1}{N} \sum_{k=0}^{N-1} f^n_k \exp(-ikj\Delta x),$$

de telle sorte que $\hat{f}^n_{k+p} = \hat{f}^n_k e^{ikp\Delta x}$. Le schéma numérique devient alors dans l'espace de Fourier

$$\hat{f}^{n+1}_k = h_k \hat{f}^n_k,\ \text{avec}\ h_k = 1 - \frac{a\Delta t}{6\Delta x} i \left(6\sin(k\Delta x) - \sin(2k\Delta x)\right).$$

Puisque $\Delta t > 0$ et $k \neq 0$, le facteur d'amplification $|h_k|$ est strictement supérieur à 1, nous voyons que ce schéma est instable.
Il est à noter que l'utilisation de schémas de Runge-Kutta d'ordre élevé permet de surmonter ce manque de stabilité des schémas de Runge-Kutta d'ordre bas (voir [32]). Notons que le schéma RK2 est instable, ce qui n'est généralement pas le cas pour les schémas semi-Lagrangiens.
En considérant les cases stables CD2 RK4 et CD4 RK4, nous cherchons à déterminer numériquement l'ordre de la méthode. Nous choisissons la condition initiale périodique $f^0(x) = \sin(2\pi x)$ sur le domaine 1D $[0, 1]$, avec les paramètres suivants :

$$\begin{cases} a = 1 \\ \Delta t = 0.001 \\ t_{\max} = 16, \end{cases} \tag{4.1.4}$$

Sur la Figure 4.1, nous traçons en rouge l'erreur en norme L^1 de la reconstruction obtenue pour différents nombres de points en espace avec la méthode CD2 RK4 (à gauche) et la méthode CD4 RK4 (à droite). Ainsi, nous voyons que CD2 RK4 est d'ordre 2 et CD4 RK4 est d'ordre 4.

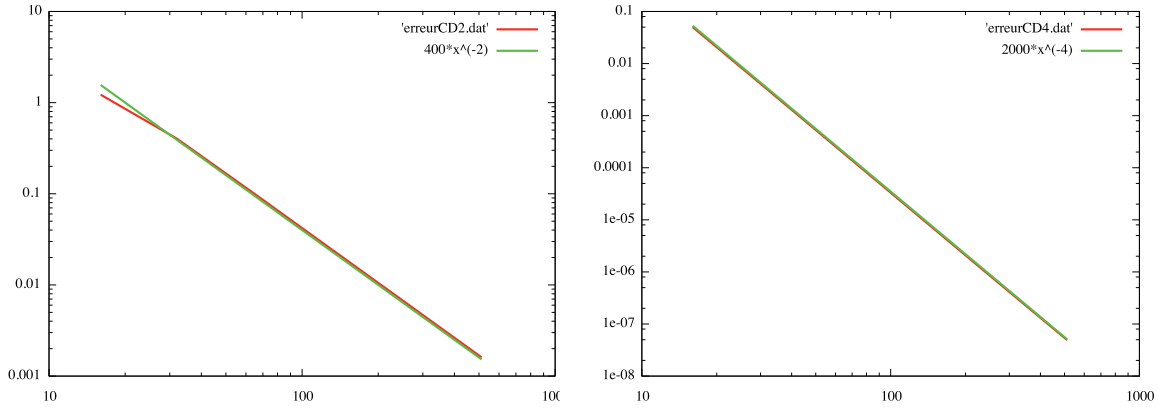


FIGURE 4.1 – Norme L^1 de l'erreur pour l'advection linéaire en fonction de N_x avec (à gauche) le schéma CD2 RK4 et (à droite) le schéma CD4 RK4. $\Delta t = 0.001$ et $t_{\max} = 16$.

4.1.3 Advection 2D

L'extension au cas 2D est détaillée ici en vue d'applications pour le système de Vlasov-Poisson. Le modèle général que nous avons à l'esprit est

$$\partial_t f(t, x, y) + \partial_x(a_x(t, x, y) f(t, x, y)) + \partial_y(a_y(t, x, y) f(t, x, y)) = 0, \tag{4.1.5}$$

avec $a = (a_x, a_y)$ un champ de vecteurs qui satisfait à la condition de divergence $\nabla \cdot a = \partial_x a_x + \partial_y a_y = 0$. Les inconnues sont alors

$$\bar{f}^n_{i,j} = \frac{1}{\Delta x \Delta y} \int_{x_{i-1/2}}^{x_{i+1/2}} \int_{y_{j-1/2}}^{y_{j+1/2}} f(t_n, x, y) dx dy.$$

Pour mettre l'accent sur la discrétisation spatiale, nous introduisons

$$\bar{f}_{i,j}(t) = \frac{1}{\Delta x \Delta y} \int_{x_{i-1/2}}^{x_{i+1/2}} \int_{y_{j-1/2}}^{y_{j+1/2}} f(t,x,y) dx dy.$$

En intégrant (4.1.5) sur un volume de contrôle et en divisant par son volume $\Delta x \Delta y$, nous obtenons :

$$\begin{aligned} \frac{d\bar{f}_{i,j}(t)}{dt} = &-\frac{1}{\Delta x \Delta y} \int_{y_{j-1/2}}^{y_{j+1/2}} \left[a_x(t, x_{i+1/2}, y) f(t, x_{i+1/2}, y) - a_x(t, x_{i-1/2}, y) f(t, x_{i-1/2}, y) \right] dy \\ &- \frac{1}{\Delta x \Delta y} \int_{x_{i-1/2}}^{x_{i+1/2}} \left[a_y(t, x, y_{j+1/2}) f(t, x, y_{j+1/2}) - a_y(t, x, y_{j-1/2}) f(t, x, y_{j-1/2}) \right] dy. \quad (4.1.6) \end{aligned}$$

Nous utilisons ensuite une formule qui permet d'exprimer l'intégrale du produit en termes d'un produit d'intégrales

Proposition 4.1.1. *Nous avons*

$$\begin{aligned} &\frac{1}{h} \int_{z_{i-1/2}}^{z_{i+1/2}} b(x) g(x) dx = \frac{1}{h} \int_{z_{i-1/2}}^{z_{i+1/2}} b(x) dx \cdot \frac{1}{h} \int_{z_{i-1/2}}^{z_{i+1/2}} g(x) dx \\ &+ \frac{1}{48 h^2} \left(\int_{z_{i+1/2}}^{z_{i+3/2}} b(x) dx - \int_{z_{i-3/2}}^{z_{i-1/2}} b(x) dx \right) \left(\int_{z_{i+1/2}}^{z_{i+3/2}} g(x) dx - \int_{z_{i-3/2}}^{z_{i-1/2}} g(x) dx \right) + O(h^4). \end{aligned}$$

Remarque 4.1.2. *Une telle formule (et d'autres plus générales) sont développées dans [33, 37] et permettent une approximation d'ordre élevé en espace.*

Nous donnons ici une preuve.

Preuve de la Proposition 4.1.1. Nous partons de l'approximation classique du point milieu :

$$\frac{1}{h} \int_{z_{i-1/2}}^{z_{i+1/2}} f(x) dx = f(z_i) + \frac{h^2}{24} \partial_x^2 f(z_i) + O(h^4).$$

Le membre de gauche donne

$$b(z_i) g(z_i) + \frac{h^2}{24} \left(b''(z_i) g(z_i) + 2 b'(z_i) g'(z_i) + b(z_i) g''(z_i) \right) + O(h^4),$$

alors que le membre de droite donne

$$\begin{aligned} &\left(b(z_i) + \frac{h^2}{24} b''(z_i) \right) \left(g(z_i) + \frac{h^2}{24} g''(z_i) \right) + O(h^4) \\ &+ \frac{1}{48} \left(b(z_{i+1}) - b(z_{i-1}) + \frac{h^2}{24} (b''(z_{i+1}) - b''(z_{i-1})) \right) \left(g(z_{i+1}) - g(z_{i-1}) + \frac{h^2}{24} (g''(z_{i+1}) - g''(z_{i-1})) \right) \\ &= b(z_i) g(z_i) + \frac{h^2}{24} \left(b''(z_i) g(z_i) + b(z_i) g''(z_i) \right) + O(h^4) \\ &\quad + \frac{h^2}{48} \left(2 b'(z_i) + O(h^2) \right) \left(2 g'(z_i) + O(h^2) \right), \end{aligned}$$

ce qui donne le résultat □

Enfin, nous procédons comme dans le cas 1D. A titre d'exemple, l'approximation CD4 donne

$$\frac{1}{\Delta y}\int_{y_{j-1/2}}^{y_{j+1/2}} f(t,x_{i+1/2},y)dy \approx \frac{7}{12}(\bar{f}_{i,j}(t)+\bar{f}_{i+1,j}(t)) - \frac{1}{12}(\bar{f}_{i-1,j}(t)+\bar{f}_{i+2,j}(t)),$$

On obtient ainsi un système d'EDO qui est discrétisé en temps avec un schéma de Runge-Kutta comme dans le cas 1D.

4.1.4 Application au système de Vlasov-Poisson

Dans le cas du système de Vlasov-Poisson, nous avons $a_x(t,x,y) = y$ et $a_y(t,x,y) = E(t,x)$ dans (4.1.5). Le champ électrique est calculé en utilisant la densité de charge $\int f(t,x,v)dv$ qui est recalculée après chaque itération de la méthode de Runge-Kutta.

4.2 Méthodes basées sur les points de Gauss en temps

Le but de cette section est de présenter une méthode de type volumes finis basée sur une intégration semi-Lagrangienne des flux. La méthode est d'abord présentée en 1D et dans ce contexte une analyse de stabilité est effectuée. Ensuite, nous aborderons le cas 2D.

4.2.1 L'équation d'advection linéaire 1D

Commençant avec l'équation d'advection (4.1.2), l'intégration en temps entre t^n et t^{n+1} conduit au calcul de $\int_{t^n}^{t^{n+1}} f(t,x_{i+1/2})dt$. Grâce au changement de variable $t = t^n + \Delta t(1+s)/2$ avec $s \in [-1,1]$, une quadrature de Gauss peut être réalisée : en introduisant les K points de Gauss et leurs poids (ω_k, τ_k) sur l'intervalle $[-1,1]$, cela conduit à

$$\int_{t^n}^{t^{n+1}} f(t,x_{i+1/2})dt \approx \frac{\Delta t}{2}\sum_{k=1}^{K}\omega_k f\left(t^n + \frac{\Delta t}{2}(1+\tau_k), x_{i+1/2}\right).$$

En utilisant le fait que f est constante le long des caractéristiques, le membre de droite peut être exprimé en fonction de $f(t^n)$

$$\int_{t^n}^{t^{n+1}} f(t,x_{i+1/2})dt \approx \frac{\Delta t}{2}\sum_{k=1}^{K}\omega_k f\left(t^n, x_{i+1/2} - a\left(\frac{\Delta t}{2}(1+\tau_k)\right)\right),$$

de telle sorte que le schéma numérique est

$$\bar{f}_i^{n+1} = \bar{f}_i^n - \frac{a\Delta t}{2\Delta x}\sum_{k=1}^{K}\omega_k\left[f\left(t^n, x_{i+1/2} - \frac{a\Delta t}{2}(1+\tau_k)\right) - f\left(t^n, x_{i-1/2} - \frac{a\Delta t}{2}(1+\tau_k)\right)\right]. \tag{4.2.1}$$

Les quantités $f(t^n, x_{i+1/2} - a\Delta t(1+\tau_k)/2)$ doivent être reconstruites à partir des valeurs moyennes connues $\bar{f}_i^n$, $i = 0,\ldots,N-1$ en utilisant un opérateur d'interpolation. Certaines reconstructions seront détaillées ci-après.

Remarque 4.2.1. *Par exemple, si nous choisissons $N_k = 1$ points de Gauss, $\omega_k = 2$ et $\tau_k = 0$, nous obtenons la formule du point milieu :*

$$\int_{t^n}^{t^{n+1}} f(t, x_{i+1/2})dt \approx \Delta t f\left(t^n, x_{i+1/2} - a\frac{\Delta t}{2}\right). \tag{4.2.2}$$

Si nous choisissons $N_k = 2$ points de Gauss, $\omega_1 = \omega_2 = 1$, $\tau_1 = -1/\sqrt{3}$ et $\tau_2 = 1/\sqrt{3}$, nous obtenons :

$$\int_{t^n}^{t^{n+1}} f(t, x_{i+1/2})dt \approx \frac{\Delta t}{2}\left(f\left(t^n, x_{i+1/2} - a\frac{\Delta t(3-\sqrt{3})}{6}\right) + f\left(t^n, x_{i+1/2} - a\frac{\Delta t(3+\sqrt{3})}{6}\right)\right). \tag{4.2.3}$$

Remarque 4.2.2. *L'introduction de points de Gauss en temps pour l'advection linéaire n'est pas vraiment utile, puisque nous avons la relation entre l'intégration en temps et l'intégration en espace (voir la Section 4.3 pour une preuve)*

$$a\int_{t_n}^{t_{n+1}} f(t, x_{i+1/2})dt = \int_{x^*_{i+1/2}}^{x_{i+1/2}} f(t_n, y)dy, \tag{4.2.4}$$

où $x^*_{i+1/2} = x_{i+1/2} - a\Delta t$ est le pied de la caractéristique s'arrêtant à $x_{i+1/2}$. Puisque les valeurs $\bar{f}^n_i$, $i = 0, \ldots, N-1$ sont connues, le membre de droite peut être approché avec une reconstruction appropriée tel que décrit après. En particulier, il n'y a alors aucune restriction de type CFL.

Cependant l'extension au cas 2D implique le calcul de l'intersection entre le maillage Lagrangien et Cartésien, voir [46]. L'utilisation des points de Gauss évite de faire cette étape technique et est donc une alternative que nous proposons d'explorer ici. D'autres stratégies peuvent également être envisagées (voir [76, 41] où les extensions possibles des volumes finis 1D aux schémas 2D non splittés sont détaillées).

Reconstruction La méthode doit être complétée par un opérateur de reconstruction pour calculer $f(t^n, x_{i+1/2} - a\Delta t(1+\tau_k)/2)$. Beaucoup d'opérateurs d'interpolation peuvent être considérés pour répondre à cette tâche [36], [87], [79]. Nous cherchons un polynôme P_i sur chaque cellule $[x_{i-1/2}, x_{i+1/2}]$ qui satisfait

$$\frac{1}{\Delta x}\int_{x_{i-1/2}}^{x_{i+1/2}} P_i(x)dx = \bar{f}^n_i. \tag{4.2.5}$$

Les reconstructions de Lagrange (LAG-$2d+1$) consistent à prendre P_i de degré $\leq 2d$ satisfaisant les contraintes

$$\frac{1}{\Delta x}\int_{x_{j-1/2}}^{x_{j+1/2}} P_i(x)dx = \bar{f}^n_i, \ j = i-d, \ldots, i+d.$$

Les reconstructions de type PPM consistent à prendre P_i de degré ≤ 2 satisfaisant (4.2.5) ainsi que

$$P_i(x_{i-1/2}) = f^n_{i-1/2}, \ P_i(x_{i+1/2}) = f^n_{i+1/2},$$

K	LAG-1	LAG-3	LAG-5	LAG-7	PPM 0	PPM 1	PPM 2
1	1.00	0.68	0.00	0.00	0.72	0.66	0.66
2	1.00	1.00	1.00	1.00	1.63	1.70	1.73
3	1.00	1.00	1.00	1.00	2.00	1.54	1.54
4	1.00	1.00	1.00	1.00	1.77	1.83	1.88
5	1.85	2.00	2.00	2.00	2.69	2.69	2.69

TABLE 4.2 – Conditions CFL avec K points de Gauss

c'est-à-dire

$$P_i(x_{i-1/2}+\alpha\Delta x) = (3\alpha^2-4\alpha+1)f^n_{i-1/2} + (3\alpha^2-2\alpha)f^n_{i+1/2} + (6\alpha-6\alpha^2)\bar{f}^n_i \quad \text{avec } \alpha\in[0,1].$$

Les valeurs aux interfaces $f^n_{i+1/2} \approx f(t^n, x_{i+1/2})$ sont données par

$$\begin{aligned}
PPM0: \quad f^n_{i+1/2} &= \frac{1}{2}(\bar{f}^n_i + \bar{f}^n_{i+1}),\\
PPM1: \quad f^n_{i+1/2} &= \frac{7}{12}(\bar{f}^n_i + \bar{f}^n_{i+1}) - \frac{1}{12}(\bar{f}^n_{i-1} + \bar{f}^n_{i+2}),\\
PPM2: \quad f^n_{i+1/2} &= \frac{37}{60}(\bar{f}^n_i + \bar{f}^n_{i+1}) - \frac{8}{60}(\bar{f}^n_{i-1} + \bar{f}^n_{i+2}) + \frac{1}{60}(\bar{f}^n_{i-2} + \bar{f}^n_{i+3}).
\end{aligned}$$

Remarque 4.2.3. *La notation LAG-$2d+1$ pourrait être étrange, puisque nous considérons des polynômes de degré $\leq 2d$. Cependant, si nous considérons la reconstruction sans l'approximation des points de Gauss, comme expliqué dans la Remarque 4.2.2, nous pouvons voir que cette méthode est équivalente au schéma semi-Lagrangian par points avec une interpolation de Lagrange de degré $\leq 2d+1$ (voir [87]).*

Analyse de stabilité et ordre
Comme dans la Sous-Section 4.1.2, nous cherchons les conditions CFL, en étudiant numériquement le facteur d'amplification. Les résultats sont donnés dans la Table 4.2.
Sur la Figure 4.2, l'erreur spatiale en norme L^1 est tracée dans le cas de l'advection constante (avec une condition initiale Gaussienne). Dans le cas de 2 points de Gauss en temps, nous voyons que les ordres sont retrouvés : LAG-3 est d'ordre 3 et LAG-5 est d'ordre 5.

4.2.2 Advection 2D

L'extension au cas 2D est discutée ici. Comme précédemment, nous intégrons sur un volume de contrôle et nous le divisons par son volume $\Delta x\Delta y$, et nous obtenons le schéma semi-discret suivant :

$$\begin{aligned}
\frac{d\bar{f}_{i,j}(t)}{dt} = &-\frac{1}{\Delta x\Delta y}\int_{y_{j-1/2}}^{y_{j+1/2}} (a_x(t,x_{i+1/2},y)f(t,x_{i+1/2},y) - a_x(t,x_{i-1/2},y)f(t,x_{i-1/2},y))dy\\
&-\frac{1}{\Delta x\Delta y}\int_{x_{i-1/2}}^{x_{i+1/2}} (a_y(t,x,y_{j+1/2})f(t,x,y_{j+1/2}) - a_y(t,x,y_{j-1/2})f(t,x,y_{j-1/2}))dy.
\end{aligned}$$

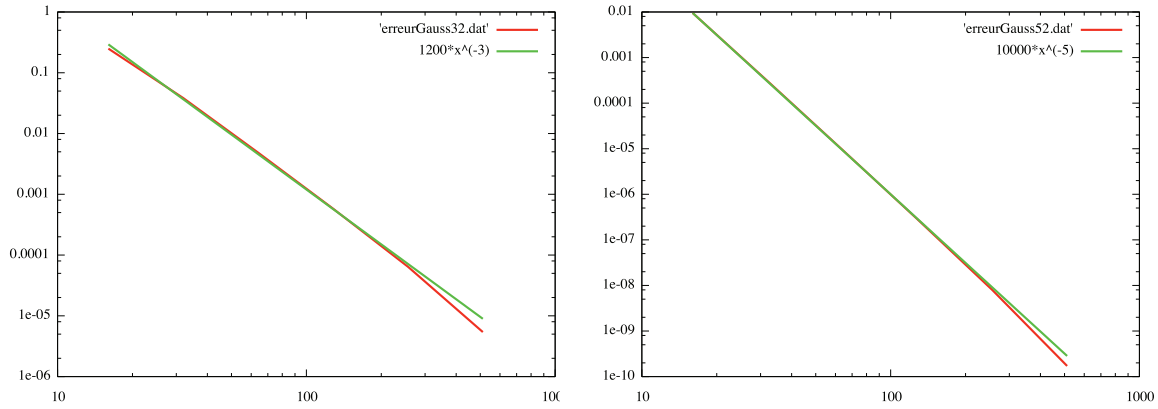


FIGURE 4.2 – Norme L^1 de l'erreur pour l'advection linéaire en fonction de N_x avec (à gauche) la reconstruction Lag-3 et (à droite) la reconstruction Lag-5. $\Delta t = 0.001$ et $t_{\max} = 16$.

Pour calculer les deux intégrales, nous introduisons ici les points de Gauss en *espace* : d_x points $(\omega_{x,\ell}, \tau_{x,\ell})$ pour la direction x, et d_y points $(\omega_{y,\ell}, \tau_{y,\ell})$ pour la direction y

$$\begin{aligned}\frac{d\bar{f}_{i,j}(t)}{dt} \approx & -\frac{1}{2\Delta x}\sum_{\ell=1}^{d_y}\omega_{y,\ell}\left[a_x\left(t, x_{i+1/2}, y_j+\frac{\Delta y}{2}(1+\tau_{y,\ell})\right) f\left(t, x_{i+1/2}, y_j+\frac{\Delta y}{2}(1+\tau_{y,\ell})\right)\right. \\ & \left. -a_x\left(t, x_{i-1/2}, y_j+\frac{\Delta y}{2}(1+\tau_{y,\ell})\right) f\left(t, x_{i-1/2}, y_j+\frac{\Delta y}{2}(1+\tau_{y,\ell})\right)\right] \\ & -\frac{1}{2\Delta y}\sum_{\ell=1}^{d_x}\omega_{x,\ell}\left[a_y\left(t, x_i+\frac{\Delta x}{2}(1+\tau_{x,\ell}), y_{j+1/2}\right) f\left(t, x_i+\frac{\Delta x}{2}(1+\tau_{x,\ell}), y_{j+1/2}\right)\right. \\ & \left. -a_y\left(t, x_i+\frac{\Delta x}{2}(1+\tau_{x,\ell}), y_{j-1/2}\right) f\left(t, x_i+\frac{\Delta x}{2}(1+\tau_{x,\ell}), y_{j-1/2}\right)\right]\end{aligned}$$

Maintenant, nous appliquons la même stratégie que dans le cas 1D en utilisant K points de Gauss en *temps* (ω_k, τ_k) pour la quadrature en temps, nous obtenons alors

$$\begin{aligned}\bar{f}_{i,j}^{n+1} \approx & \bar{f}_{i,j}^{n} - \frac{\Delta t}{4\Delta x}\sum_{k=1}^{K}\sum_{\ell=1}^{d_y}\omega_k\omega_{y,\ell}\left[a_x\left(t^n+\frac{\Delta t}{2}(1+\tau_k), x_{i+1/2}, y_j+\frac{\Delta y}{2}(1+\tau_{y,\ell})\right) f\left(t^n, \left(x_{i+1/2}, y_j+\right.\right.\right. \\ & \left.\left.\left. -a_x\left(t^n+\frac{\Delta t}{2}(1+\tau_k), x_{i-1/2}, y_j+\frac{\Delta y}{2}(1+\tau_{y,\ell})\right) f\left(t^n, \left(x_{i-1/2}, y_j+\frac{\Delta y}{2}(1+\tau_{y,\ell})\right)^{*_k}\right)\right] \right. \\ & -\frac{\Delta t}{4\Delta y}\sum_{k=1}^{K}\sum_{\ell=1}^{d_x}\omega_k\omega_{x,\ell}\left[a_y\left(t^n+\frac{\Delta t}{2}(1+\tau_k), x_i+\frac{\Delta x}{2}(1+\tau_{x,\ell}), y_{j+1/2}\right) f\left(t^n, \left(x_i+\frac{\Delta x}{2}(1+\tau_{x,\ell}), y_{j+1/2}\right.\right.\right. \\ & \left.\left.\left. -a_y\left(t^n+\frac{\Delta t}{2}(1+\tau_k), x_i+\frac{\Delta x}{2}(1+\tau_{x,\ell}), y_{j-1/2}\right) f\left(t^n, \left(x_i+\frac{\Delta x}{2}(1+\tau_{x,\ell}), y_{j-1/2}\right)^{*_k}\right)\right]\right.,\end{aligned}$$

où $(x,y)^{*_k}$ dénote le pied au temps t^n de la caractéristique se terminant en (x,y) au temps $t^n + \Delta t(1+\tau_k)/2$. En utilisant un schéma prédicteur-correcteur par exemple, nous supposons que les champs a_x et a_y sont constants sur le domaine temporel $[t^n, t^{n+1}]$: $a_x(t,x,y) \simeq a_x(t^{n+1/2},x,y)$ et $a_y(t,x,y) \simeq a_y(t^{n+1/2}x,y)$ pour tout $t \in [t^n, t^{n+1}]$, et les champs $a_x(t^{n+1/2},x,y)$ et $a_y(t^{n+1/2},x,y)$ sont prévus avec une méthode appropriée.

Remarque 4.2.4. *Le cas d'un point de Gauss en espace sera essentiellement utilisé. Le schéma numérique s'écrit*

$$\bar{f}_{i,j}^{n+1} \approx \bar{f}_{i,j}^{n} - \frac{\Delta t}{2\Delta x} \sum_{k=1}^{K} \omega_k \left[a_x(t^{n+1/2}, x_{i+1/2}, y_j) f(t^n, (x_{i+1/2}, y_j)^{*_k}) - a_x(t^{n+1/2}, x_{i-1/2}, y_j) f(t^n, (x_{i-1/2}, y_j \right.$$

$$\left. - \frac{\Delta t}{2\Delta y} \sum_{k=1}^{K} \omega_k \left[a_y(t^{n+1/2}, x_i, y_{j+1/2}) f(t^n, (x_i, y_{j+1/2})^{*_k}) - a_y(t^{n+1/2}, x_i, y_{j-1/2}) f(t^n, (x_i, y_{j-1/2})^{*_k}) \right] \right., \tag{4.2.6}$$

où $(x_{i+1/2}, y_j)^{*_k}$ *dénote le pied au temps* t^n *de la caractéristique se terminant en* $(x_{i+1/2}, y_j)$ *au temps* $t^n + \Delta t(1+\tau_k)/2$.

4.2.3 Application au cas Vlasov-Poisson

Nous allons maintenant nous concentrer sur l'équation de Vlasov-Poisson qui correspond à $a_x(t,x,y) = y$ et $a_y(t,x,y) = E(t,x)$.
Dans ce cas, (4.2.6) peut être simplifié en

$$\bar{f}_{i,j}^{n+1} \approx \bar{f}_{i,j}^{n} - \frac{\Delta t v_j}{2\Delta x} \sum_{k=1}^{K} \omega_k \left[f(t^n, (x_{i+1/2}, v_j)^{*_k}) - f(t^n, (x_{i-1/2}, v_j)^{*_k}) \right]$$
$$- \frac{\Delta t E(t^{n+1/2}, x_i)}{2\Delta v} \sum_{k=1}^{K} \omega_k \left[f(t^n, (x_i, v_{j+1/2})^{*_k}) - f(t^n, (x_i, v_{j-1/2})^{*_k}) \right].$$

Prédiction de $E(t^{n+1/2})$**.** Le champ électrique $E(t^{n+1/2})$ est approximé par une prédiction : nous calculons $\bar{f}_{i,j}^{n+1/2}$ en utilisant le schéma avec $\Delta t/2$ à la place de Δt et nous considérons le champ électrique en utilisant la densité de charge au temps t^n. Ceci permet de calculer la densité de charge et donc l'approximation de $E(t^{n+1/2})$ en utilisant $\bar{f}_{i,j}^{n+1/2}$, qui est utilisé pour l'étape de correction.
Calcul des caractéristiques Un schéma de Verlet est utilisé pour le calcul des caractéristiques : en écrivant par exemple $(X^{n+1}, V^{n+1}) = (x_{i+1/2}, v_j)$ et $(X^n, V^n) = (x_{i+1/2}, v_j)^{*_k}$, nous avons

$$\begin{cases} X^{n+1/2} = X^{n+1} - \frac{\Delta t}{2} V^{n+1} \\ V^n = V^{n+1} - \Delta t E(X^{n+1/2}) \\ X^n = X^{n+1/2} - \frac{\Delta t}{2} V^n, \end{cases}$$

où E correspond soit à $E(t^n)$ (étape de prédiction) soit à $E(t^{n+1/2})$ (étape de correction).

Reconstruction 2D La reconstruction 2D qui est nécessaire ici consiste à utiliser un produit tensoriel de reconstructions 1D.

Remarque 4.2.5. *Nous avons constaté que toutes les méthodes PPM sont instables dans le cas 2D. Nous pouvons procéder à l'analyse de Von Neumann pour* $a_x = 1, a_y = 0$ *et une condition CFL* $A = \Delta t/\Delta x$*. Nous utilisons la condition initiale* $\exp(ik_x x)\exp(ik_y y)$.

En choisissant $k_x = \pi/6$ et $k_y = 4\pi/6$, nous trouvons que le facteur d'amplification h_{k_x,k_y} satisfait la formule

$$1-|h_{\pi/6,4\pi/6}|^2 = \begin{cases} (-\frac{25861743}{5120000} + \frac{746487}{256000}\sqrt{3})A^6 + (\frac{25861743}{1280000} - \frac{746487}{64000}\sqrt{3})A^5 + (-\frac{25517833}{1280000} + \frac{3682727}{320000}\sqrt{3})A^4 \\ +(\frac{166379}{80000}\sqrt{3} - \frac{231251}{64000})\,A^3 + (-\frac{1040553}{320000}\sqrt{3} + \frac{887043}{160000})\,A^2, \qquad \textit{(cas PPM2)}, \\ \\ (\frac{188307}{32768} + \frac{217413}{65536}\sqrt{3})A^6 + (-\frac{217413}{16384}\sqrt{3} + \frac{188307}{8192})\,A^5 + (\frac{213591}{16384}\sqrt{3} - \frac{184975}{8192})\,A^4 \\ +(-\frac{4193}{1024} + \frac{9639}{4096}\sqrt{3})A^3 + (\frac{25389}{4096} - \frac{931}{256}\sqrt{3})A^2, \qquad \textit{(cas PPM1)}, \\ \\ (\frac{16335}{4096}\sqrt{3} - \frac{14157}{2048})\,A^6 + (\frac{14157}{512} - \frac{16335}{1024}\sqrt{3})A^5 + (-\frac{13915}{512} + \frac{16093}{1024}\sqrt{3})A^4 \\ +(\frac{649}{256}\sqrt{3} - \frac{583}{128})\,A^3 + (-\frac{33}{8}\sqrt{3} + \frac{1815}{256})\,A^2, \qquad \textit{(cas PPM0)}. \end{cases}$$

Puisque le coefficient dominant en A^2 est toujours négatif, il existe $A_0 > 0$ tel que pour $0 < A < A_0$, la méthode est instable. Notons que ce phénomène n'apparaît pas dans le cas 1D.

4.3 Liens entre schémas de volumes finis et schémas semi-Lagrangiens

Nous établissons d'abord l'égalité (4.2.4), qui fait le lien entre les volumes finis et la formulation semi-Lagrangienne du flux. Ce résultat, valable pour un champ général $a(t,x)$, a déjà été prouvé dans [28] par exemple, en utilisant le théorème de la divergence. Nous donnons ici une preuve alternative.

Proposition 4.3.1. *Nous avons*

$$\int_{t_n}^{t_{n+1}} a(t,x_{i+1/2})f(t,x_{i+1/2})dt = \int_{x^*_{i+1/2}}^{x_{i+1/2}} f(t_n,y)dy,$$

où

$$\partial_t f(t,x) + \partial_x(a(t,x)f(t,x)) = 0,\ X'(t) = a(t,X(t)),\ X(t_{n+1}) = x_{i+1/2},\ X(t_n) = x^*_{i+1/2}.$$

Démonstration. Soit $X(t,s,x)$ la caractéristique satisfaisant $\partial_t X(t,s,x) = a(t,X(t,s,x)),\ X(s,s,x) = x$. Nous avons d'abord, en suivant [43]

$$\int_{t_n}^{t_{n+1}} a(t,x_{i+1/2})f(t,x_{i+1/2})dt = \int_{t_n}^{t_{n+1}} a(t,x_{i+1/2})f(t_n,X(t_n,t,x_{i+1/2}))\partial_x X(t_n,t,x_{i+1/2})dt.$$

Nous faisons le changement de variable $y = X(t_n,t,x_{i+1/2})$, afin de passer de l'intégrale en temps à l'intégrale en espace. Notons que nous avons

$$X(t_n,t',X(t',t,x_{i+1/2})) = X(t_n,t,x_{i+1/2}),\ \forall t',$$

ce qui signifie que la quantité ne dépend pas de t'. La dérivée en t' vaut donc zéro, ce qui signifie que

$$\partial_s X(t_n,t',X(t',t,x_{i+1/2})) + \partial_t X(t',t,x_{i+1/2})\partial_x X(t_n,t',X(t',t,x_{i+1/2})) = 0,$$

d'où

$$\partial_s X(t_n, t', X(t', t, x_{i+1/2})) = -a(t, X(t', t, x_{i+1/2}))\partial_x X(t_n, t', X(t', t, x_{i+1/2})).$$

En prenant $t' = t$, nous obtenons

$$\partial_s X(t_n, t, x_{i+1/2}) = -a(t, x_{i+1/2})\partial_x X(t_n, t, x_{i+1/2}),$$

et donc $dy = -a(t, x_{i+1/2})\partial_x X(t_n, t, x_{i+1/2})dt$. Puisque nous avons $X(t_n, t_n, x_{i+1/2}) = x_{i+1/2}$ et $X(t_n, t_{n+1}, x_{i+1/2}) = x^*_{i+1/2}$, nous avons

$$\int_{t_n}^{t_{n+1}} a(t, x_{i+1/2}) f(t, x_{i+1/2}) dt = -\int_{x_{i+1/2}}^{x^*_{i+1/2}} f(t_n, y) dy,$$

ce qui donne le résultat. Nous nous référons à [34] pour de tels calculs sur les caractéristiques. □

Intégrateur exponentiel Nous faisons un lien entre le système d'EDO (la méthode des lignes) provenant de la formulation des volumes finis (Section 4.1) et le schéma semi-Lagrangien (la limite où le nombre de points de Gauss tend vers l'infini dans la Section 4.2) pour une discrétisation donnée en espace. Nous considérons le problème d'advection constant. Nous avons la proposition suivante

Proposition 4.3.2. *Considérons le schéma semi-Lagrangien avec une reconstruction LAG-$2d+1$ appliqué M fois avec un pas de temps $\Delta t/M$ qui peut être écrit sous la forme*

$$(f_j^{n+1,M})_{j=0,\dots,N-1} = \prod_{k=1}^{M} \mathcal{T}_{\Delta t/M}(f_j^{n,M})_{j=0,\dots,N-1}.$$

Nous avons alors

$$\lim_{M\to\infty} f_j^{n,M} = \bar{f}_j(t_n),\ j = 0, \dots, N-1,$$

où $(\bar{f}_j)_{j=0,\dots,N-1}$ résout (4.1.2) en prenant l'approximation upwind UP-$(2d+1)$ (4.1.3) avec $s = -r = d$ (pour $a > 0$).

Démonstration. En considérant tout d'abord le système semi-discret de la méthode des volumes finis, nous avons

$$\frac{d\bar{f}_i}{dt} = -a(f_{i+1/2} - f_{i-1/2}),\ a > 0,$$

où les flux sont approximés par un schéma upwind $f(t, x_{i+1/2}) \approx \sum_{j=-d}^{d} a_j \bar{f}_{i+j}(t)$ avec les coefficients satisfaisant

$$\frac{1}{2^k} = \sum_{j=-d}^{d} a_j \int_{j-1/2}^{j+1/2} x^k dx,\ k = 0, \dots, 2d.$$

La solution du système d'ODE (intégrateur exponentiel) peut être considérée comme une approximation d'Euler en temps utilisant le pas de temps $\Delta t/M$ et regardant la limite

$M \to +\infty$. Ceci peut être réalisé facilement à l'aide d'une analyse de Von Neumann. En effet, avec $f_{i+1/2} = \sum_{j=-d}^{d} c_j \bar{f}_{i+j}^n$, nous avons alors $(\widehat{f}_{i+1/2})_k = \sum_{j=-d}^{d} c_j \widehat{\bar{f}}_k^n e^{\imath kj\Delta x}$ et

$$(\widehat{\bar{f}}^{n+1})_k = \widehat{\bar{f}}_k^n \left(1 - \nu \left(\sum_{j=-d}^{d} c_j (e^{\imath jk\Delta x} - e^{\imath (j-1)k\Delta x})\right)\right) = \widehat{\bar{f}}_k^n (1 - \nu h(k)),$$

où $\nu = a\Delta t/(M\Delta x)$ et $h(k)$ dénote la transformée de Fourier des flux.
Nous avons alors

$$\begin{aligned} \lim_{M\to+\infty} (1 - \nu h(k))^M &= \lim_{M\to+\infty} \exp(M \ln(1 - \nu h(k))) \\ &= \lim_{M\to+\infty} \exp(-M\nu h(k)) \\ &= \exp(-a\Delta t/\Delta x\ h(k)). \end{aligned}$$

Nous observons que l'intégrateur est donné par l'exponentielle de $-a\Delta t/\Delta x$ fois la transformée de Fourier des flux.

De l'autre côté, nous considérons la méthode semi-Lagrangienne

$$\bar{f}_i^{n+1} = \frac{1}{\Delta x} \int_{x_{i-1/2}-a\Delta t/M}^{x_{i+1/2}-a\Delta t/M} f(t^n, x)dx,$$

où $f(t^n, x)$ est reconstruit par une fonction polynomiale p_i de degré $2d$ satisfaisant les contraintes

$$\frac{1}{\Delta x} \int_{x_{i-1/2+j}}^{x_{i+1/2+j}} p_i(x)dx = \bar{f}_{i+j}^n = \frac{1}{\Delta x}(P_i(x_{i+1/2+j}) - P_i(x_{i-1/2+j})),\ j = -d, \ldots, d,$$

où P_i désigne une primitive de p_i. Ainsi, le schéma numérique s'écrit, en fonction de P_i

$$\bar{f}_i^{n+1} = \frac{1}{\Delta x}(P_i(x_{i+1/2} - a\Delta t/M) - P_i(x_{i-1/2} - a\Delta t/N)),$$

où nous avons supposé $a > 0$ et $a\Delta t/M < \Delta x$. Avec un développement de Taylor de $P_i(x_{i+1/2} - a\Delta t/M)$, nous obtenons

$$\bar{f}_i^{n+1} = \bar{f}_i^n - \nu(p_i(x_{i+1/2}) - p_i(x_{i-1/2})) + O(\nu^2),$$

avec $\nu = a\Delta t/(M\Delta x)$. Une analyse de Von Neumann conduit à $(\widehat{\bar{f}}^{n+1})_k = (\widehat{\bar{f}}^n)_k(1 - \nu h(k) + O(\nu^2))$ où $h(k)\widehat{\bar{f}}_k^n$ dénote la transformée de Fourier de $[p_i(x_{i+1/2}) - p_i(x_{i-1/2})]$. Ainsi, regarder la limite $M \to +\infty$ conduit à

$$\lim_{M\to+\infty} (1 - \nu h(k) + O(\nu^2))^M = \lim_{M\to+\infty} \exp(M \ln(1 - \nu h(k) + O(\nu^2))) = \exp(-a\Delta t/\Delta x\ h(k)).$$

Il suffit alors de prouver que $p_i(x_{i+1/2})$ (dans la méthode conservative) est égale à l'approximation de $f(x_{i+1/2})$ (dans la méthode par volumes finis). La valeur $p_i(x_{i+1/2})$ peut être écrite comme $p_i(x_{i+1/2}) = \sum_{j=-d}^{d} a_j \bar{f}_{i+j}^n$ où a_j satisfait le système de Vandermonde, qui correspond bien à l'approximation des flux $f(x_{i+1/2})$ obtenus par la méthode des volumes finis. □

Remarque 4.3.3. *Une correspondance similaire peut être établie pour les schémas aux différences centrées (CD). En particulier, les analogues de CD2, CD4 et CD6 de la Section 4.1 sont PPM0, PPM1 et PPM2 de la section 4.2.*

Remarque 4.3.4. *Pour les schémas semi-Lagrangiens, nous pouvons également utiliser les approximations upwind dans la reconstruction au lieu de PPM :*

$$P_i(x_{i-1/2}+\alpha\Delta x) = (3\alpha^2-4\alpha+1)f^n_{(i-1/2)^+} + (3\alpha^2-2\alpha)f^n_{(i+1/2)^-} + (6\alpha-6\alpha^2)\bar{f}^n_i \quad \text{avec } \alpha\in[0,1],$$

et $f^n_{(i+1/2)^+}$ (resp. $f^n_{(i+1/2)^-}$) est reconstruit en utilisant (4.1.3) avec $s=d+1, r=-d+1$ (resp. $s=-r=d$). Dans le cas $d=0,1$, ce schéma coïncide avec LAG-$(2d+1)$. Pour d plus grand, il ne coïncide pas avec LAG-$2d+1$ (puisque la reconstruction est toujours de degré 3 donc pas de même degré que LAG-$(2d+1)$). Cependant, la limite de l'"intégrateur exponentiel" (tel que défini dans la Proposition 4.3.2) coïncide. En particulier, nous pouvons obtenir un meilleur ordre de précision à la limite (voir aussi [23]).

Remarque 4.3.5. *Nous pouvons vérifier que les schémas CD préservent exactement la norme L^2 discrète $\sum_{j=0}^{N-1}|f_j(t)|^2$. De l'autre côté, les schémas upwind font diminuer la norme L^2 : nous pouvons vérifier que*

$$\sum_{j=-d}^{d} a_j(\cos(j\omega)-\cos((j-1)\omega)) \geq 0, \quad 0\leq\omega\leq 2\pi,$$

pour $d=3$ par exemple, et cette relation reste vraie d'après la stabilité du schéma LAG-$2d+1$, pour tout $d\in\mathbb{N}$. La conservation de la norme L^2 qui est à première vue une bonne propriété n'est pas satisfaisante, car elle peut généralement conduire à des oscillations parasites. Au contraire, peu de dissipation, obtenue avec une approximation upwind d'ordre élevé des dérivés semble meilleure au régime limite. Voir aussi [11], pour les schémas d'interpolation pairs et impairs. Notons que ceci est un point clé dans [33] ; ici, un schéma non linéaire est considéré : une approximation centrée est utilisée là où la solution est régulière et une approximation upwind d'un degré inférieur est utilisé là où la solution n'est pas régulière. Nous pouvons également remarquer que lorsque de relativement grands pas de temps sont utilisés, la norme L^2 diminue généralement dans un schéma semi-Lagrangien avec une reconstruction centrée des dérivés (par exemple les splines cubiques, PPM) et cela peut empêcher des oscillations parasites, que l'on observe dans le cas des volumes finis.

Remarque 4.3.6. *Nous pourrions nous étonner de l'existence d'une discrétisation en temps pour le schéma de volumes finis telle qu'elle coïncide avec un schéma semi-Lagrangien, au moins pour $|a|\Delta t\leq\Delta x$. Ceci peut être réalisé avec une procédure de Cauchy-Kovalevsky [48], comme indiqué dans [11].*

Remarque 4.3.7. *Nous n'avons pas spécifié comment calculer la condition initiale $\bar{f}^0_j$, $j=0,\ldots,N-1$. Puisque nous nous intéressons à des volumes finis, un choix naturel pourrait être d'utiliser*

$$\bar{f}^0_j = \frac{1}{\Delta x}\int_{x_{j-1/2}}^{x_{j+1/2}} f(0,x)dx.$$

Cependant, avec ce choix nous perdons l'ordre élevé de l'approximation. Nous utiliserons plutôt l'approximation du point milieu

$$\bar{f}^0_j = f(0,x_j),$$

ce qui conduit à une précision d'ordre élevé, puisque, dans le contexte semi-Lagrangien, le schéma est alors équivalent au schéma semi-Lagrangian par point, comme indiqué dans [87]. Dans [47], les auteurs classifient ce type de méthodes dans les schémas semi-Lagrangiens aux différences finies (et non volumes finis) et présentent ce type de schémas en introduisant une fonction h satisfaisant

$$f(t_n, x_j) \simeq \frac{1}{\Delta x} \int_{x_{j-1/2}}^{x_{j+1/2}} h(t_n, x) dx,$$

qui est ensuite mise à jour à la manière des volumes finis. Nous soulignons que l'équivalence n'est valable que pour le cas de l'advection constante. Voir [47] pour les détails et d'autres reconstructions similaires, dans le contexte WENO.

4.4 Résultats numériques

Cette section est consacrée à la présentation des résultats numériques des différents schémas décrits ci-dessus. Nous nous concentrons sur les cas de test de l'équation de Vlasov-Poisson. Nos résultats seront comparés aux méthodes semi-Lagrangiennes de référence "Lag3" et "Lag5" (voir [43, 87] pour plus de détails).

Pour la méthode VFSL2, nous allons considérer deux points de Gauss en temps et l'algorithme de Verlet pour la recherche des pieds des caractéristiques. la reconstruction est réalisée avec Lagrange 3 et 5. Deux versions sont alors prises en compte, avec ou sans splitting. Ces méthodes seront appelées Vfsl3 et Vfsl5 pour l'approche avec splitting et Vfsl3-ns, Vfsl5-ns pour l'approche sans splitting. Certains résultats seront également présentés en utilisant une reconstruction PPM1 avec une procédure de splitting.

Pour la méthode de volumes finis, nous présentons les résultats pour CD4 et UP5 avec une intégration en temps RK4. Notez que dans nos cas test, le dernier terme de la Proposition 4.1.1 n'affecte pas de manière significative les résultats numériques qui sont présentés ici sans cette correction.

Deux cas test sont étudiés, le cas test Bump On Tail (BOT) présenté dans [49] et l'instabilité double faisceaux (two stream instability : TSI) (voir [87]).

4.4.1 Bump on tail

La condition initiale s'écrit

$$f_0(x, y) = \left(\frac{9}{10\sqrt{2\pi}} \exp\left(-\frac{y^2}{2}\right) + \frac{2}{10\sqrt{2\pi}} \exp(-2(y-4.5)^2) \right) (1 + 0.03\cos(0.3x))$$

où $(x, y) \in [0, L] \times [-9, 9], L = 20\pi$.

Nous considérons les paramètres numériques suivants : $N_x = N_v = 128$, $\Delta t = 0.01$. Le modèle Vlasov-Poisson (3.7.1) préserve des quantités physiques en temps qui seront utilisées pour comparer les méthodes. Premièrement, nous regardons l'évolution en temps des normes L^p de f $(p = 1, 2)$, mais également l'énergie totale $\mathcal{E}$ du système, qui est la somme de l'énergie cinétique $\mathcal{E}_k$ et de l'énergie électrique $\mathcal{E}_e$

$$\mathcal{E}(t) = \mathcal{E}_k(t) + \mathcal{E}_e(t) = \int_0^L \int_{\mathbb{R}} f(t, x, y) \frac{y^2}{2} dy dx + \frac{1}{2} \int_0^L E^2(t, x) dx.$$

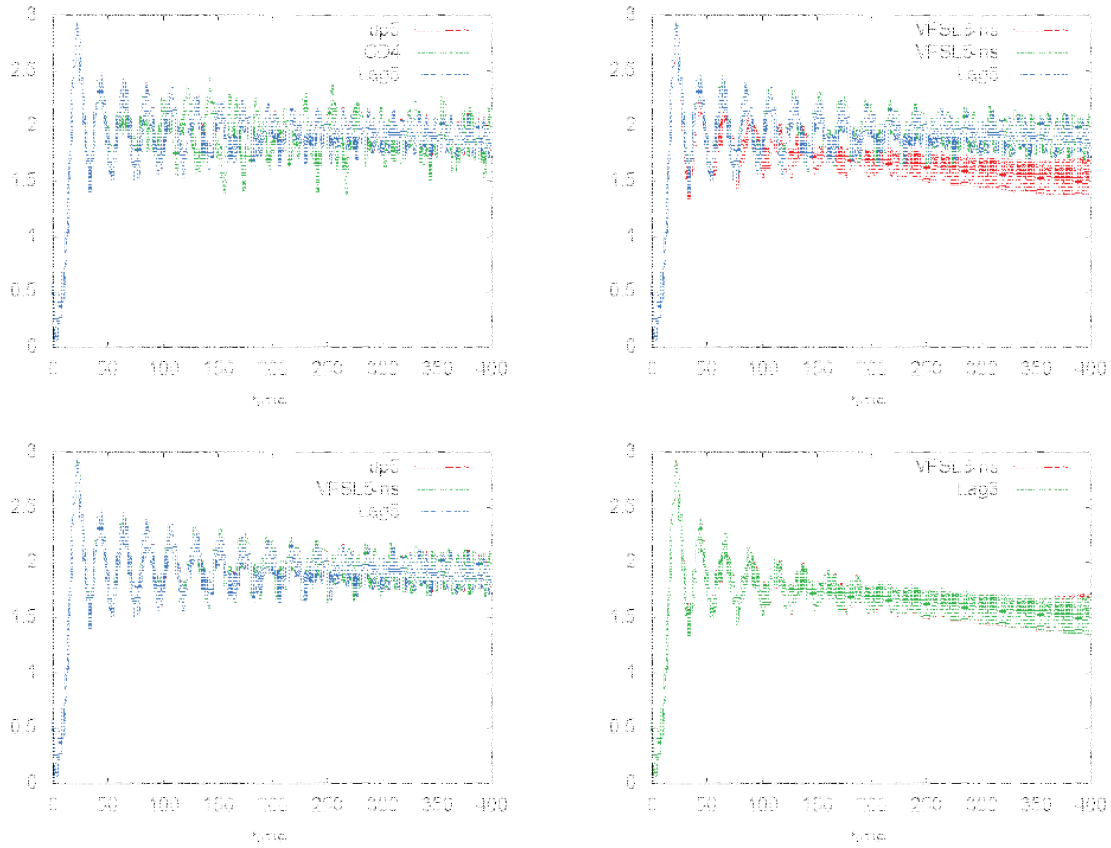


FIGURE 4.3 – Cas test Bump On Tail : évolution en temps de l'énergie électrique pour les méthodes de "Banks" (CD4 et up5), pour les méthodes Vfsl non splittées (Vfsl3-ns et Vfsl5-ns) et pour la méthode demi-Lagrangienne (Lag5). $N_x = N_v = 128$, $\Delta t = 0.01$.

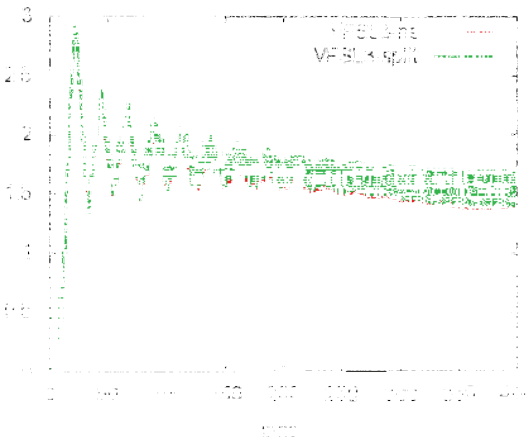


FIGURE 4.4 – Cas test Bump On Tail : évolution en temps de l'énergie électrique pour les méthodes Vfsl (méthodes Vfsl non-splitées (Vfsl3-ns et Vfsl5-ns) et splitté une fois pour la méthode semi-Lagrangienne (Lag5). $N_x = N_v = 128$, $\Delta t = 0.01$.

Sur la figure 4.3, nous traçons l'évolution en temps de l'énergie électrique. Tout d'abord, nous pouvons observer le très bon comportement de toutes les méthodes en ce qui concerne ce diagnostic. L'énergie électrique augmente au début (phase linéaire) et présente un comportement oscillant en temps long. Elle se réfère à un équilibre de type BGK composé de trois sommets qui se déplacent à vitesse intitiale $v_t = 4.5$. Nous remarquons aussi le fait que up5, Lag5 et Vfsl5 sont très similaires. En effet Vfsl5 et Lag5 ont la même reconstruction ; pour up5 et Lag5, le lien a été expliqué dans la Proposition 4.3.2. Clairement, la même chose est vraie pour méthodes de reconstruction de d'ordre 3 Vfsl3-ns et Lag3.

Notons également le comportement diffusif de la méthode basée sur du Lagrange d'ordre 3 (Vfsl3-ns et Lag3) qui a été exposé dans [87] ; lorsque des structures fines se développent, elles sont éliminées plus rapidement que lorsqu'une reconstruction d'ordre plus élevé est utilisée (méthodes basées sur du Lagrange 5 comme up5, Lag5, Vfsl5-ns). Par conséquent, le comportement en temps long de l'énergie électrique est meilleur.

Sur la figure 4.4, nous comparons la version splittée et non-splittée de Vfsl3. Nous pouvons observer que les deux versions sont très similaires ce qui valide notre approche. En effet, dans le contexte de l'équation de Vlasov-Poisson, la procédure de splitting peut être utilisée et peut être considérée comme une solution de référence. Une figure similaire est obtenue pour Vfsl5.

Sur la Figure 4.5, nous traçons l'évolution en temps de l'énergie totale pour différentes méthodes. Exceptée la méthode basée sur Lagrange 3, nous observons que cette quantité est très bien conservée. Notons que cette conservation est très difficile à obtenir et l'utilisation de reconstruction d'ordre élevé permet d'obtenir un bon comportement de l'énergie totale.

Sur la figure 4.6, nous nous sommes intéressés à l'évolution en temps de la norme L^2. Nous avons observé que CD4 préserve très bien la norme L^2 tandis que dans les autres cas, cette quantité diminue dans le temps. Pour les méthodes basées sur Lagrange 5, après la diminution autour de $t = 50$ (ce qui correspond à un moment de la création de structures qui sont plus petites que la taille de la grille et sont ensuite éliminées par le schéma), nous pouvons observer que la norme L^2 est presque constante, ce qui n'est pas le cas des méthodes basées sur Lagrange 3. Cela motive également l'utilisation de reconstructions d'ordre élevé.

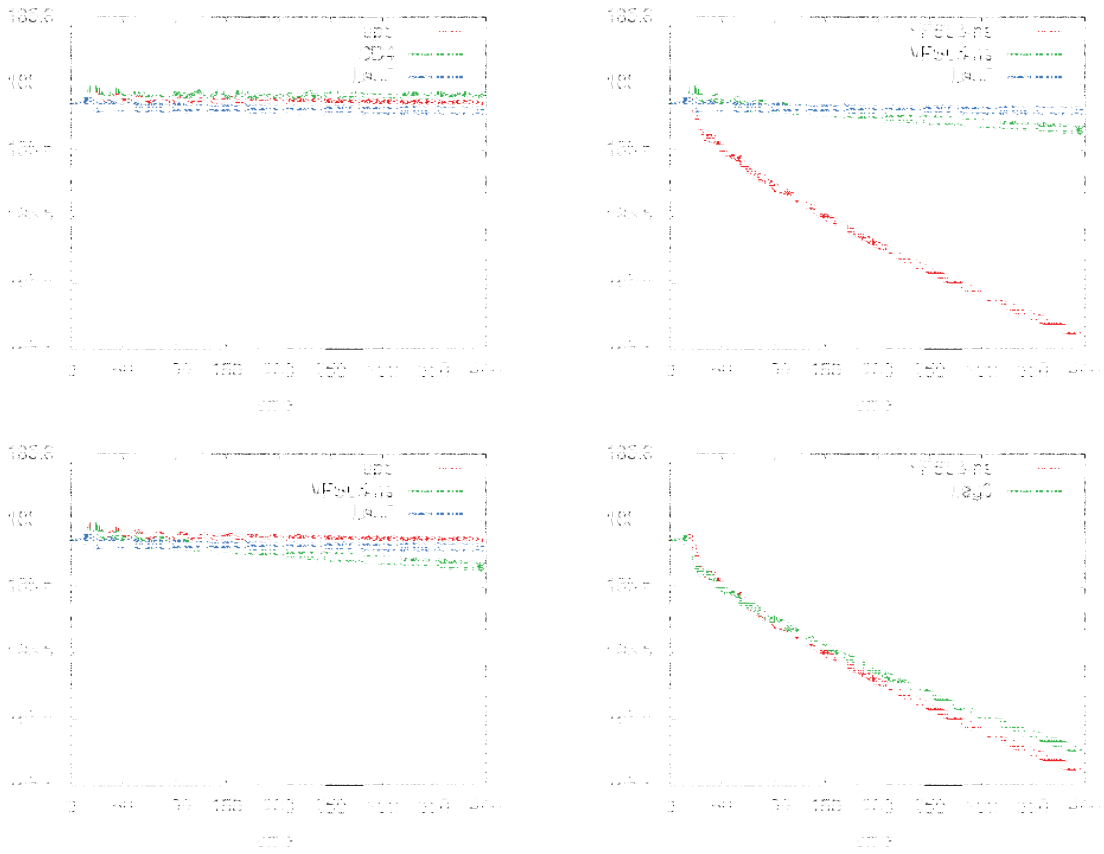


FIGURE 4.5 – Cas test Bump On Tail : évolution en temps de l'énergie totale pour la méthode de "Banks" (CD4 et up5), pour la méthode Vfsl sans splitting (Vfsl3-ns et Vfsl5-ns) et pour la méthode semi-Lagrangienne (Lag3 et Lag5). $N_x = N_v = 128$, $\Delta t = 0.01$.

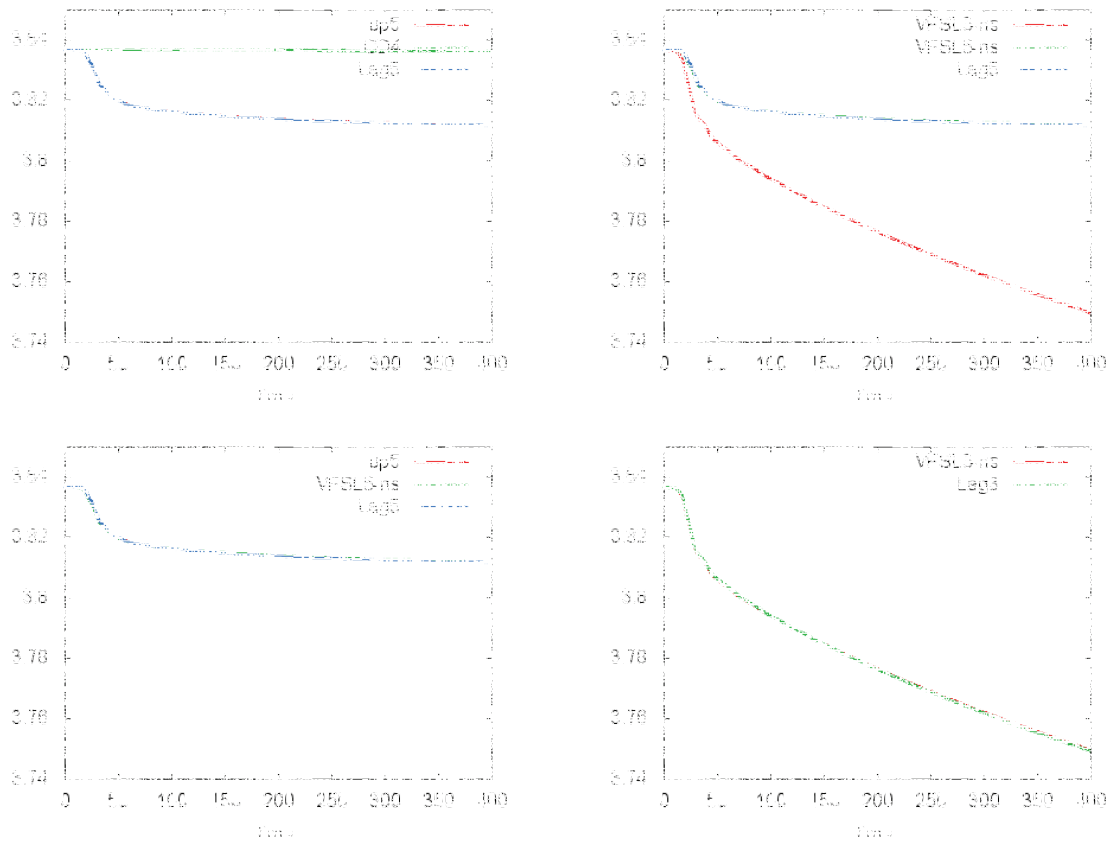


FIGURE 4.6 – Cas test Bump On Tail : évolution en temps de la norme L^2 pour la méthode de "Banks" (CD4 et up5), pour la méthode Vfsl sans splitting (Vfsl3-ns et Vfsl5-ns) et pour la méthode semi-Lagrangienne (Lag3 et Lag5). $N_x = N_v = 128$, $\Delta t = 0.01$.

Sur la Figure 4.7, l'évolution de la norme L^1 est tracée. Ce que nous voyons ici est une tendance contraire du diagnostic précédent : CD4 présente un très mauvais comportement en ce qui concerne la positivité par rapport aux autres méthodes. En effet, comme mentionné dans [33], CD4 présente des oscillations qui ne peuvent pas être détectées par le diagnostic de la norme L^2, mais qui sont mises en évidence sur le diagnostic de la norme L^1. Nous pouvons également observer les résultats des méthodes up5, Vfsl5-ns et Lag5 qui sont très proches.

4.4.2 Instabilité double faisceaux

La condition initiale est donnée par

$$f_0(x,y) = \frac{2}{7\sqrt{2\pi}}(1+5y^2)e^{-\frac{y^2}{2}}(1+0.01(\cos(0.5x)+(\cos(x)+\cos(1.5x)))/1.2),$$

avec $(x,y) \in [0,4\pi] \times [-6,6]$. Nous considérons les paramètres numériques suivants : $N_x = N_v = 128$ et $\Delta t = 0.005$. Nous présentons ici les diagnostics 2D de la fonction de distribution complète.

Les résultats proposés dans la figure 4.8 confirment les observations du cas-test précédent. En effet, CD4 présente un grand nombre d'oscillations qui conduisent à une mauvaise qualité.

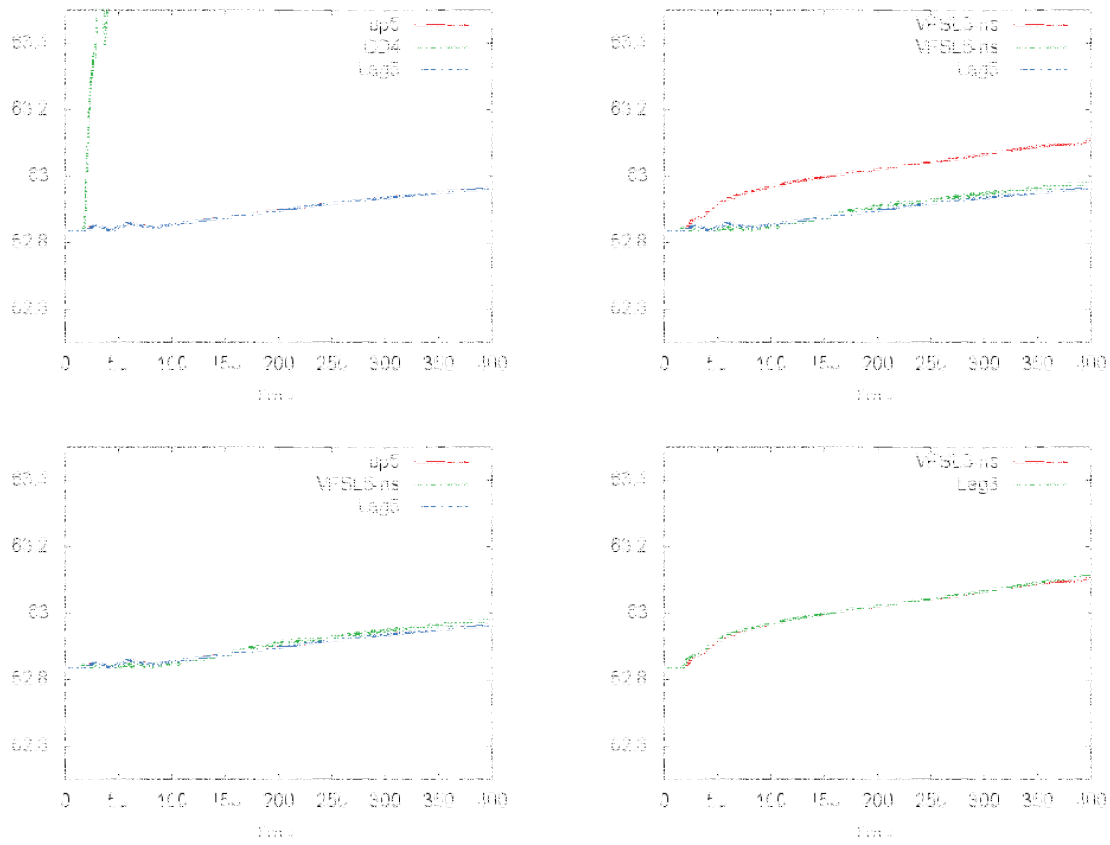


FIGURE 4.7 – Cas test Bump On Tail : évolution en temps de la norme L^1 pour la méthode de "Banks" (CD4 et up5), pour la méthode Vfsl sans splitting (Vfsl3-ns et Vfsl5-ns) et pour la méthode semi-Lagrangienne (Lag3 et Lag5). $N_x = N_v = 128$, $\Delta t = 0.01$.

En outre, l'utilisation d'une reconstruction d'ordre 3 (comme pour Vfsl3-ns) conduit à une solution très lisse ; quand elle est comparée à une solution de référence (tracée sur la Figure 4.9), nous pouvons voir que les détails ont été éliminés par le schéma. Quand un ordre plus élevé est utilisé (comme pour up5 ou Vfsl5-ns), de petites structures supplémentaires sont décrites. Sur les figures 4.10, 4.11, nous voyons à nouveau le lien entre LAG3 et up3, LAG5 et up5 et également CD4 et PPM1 pour les petites valeurs de Δt comme le montre la Remarque 4.3.3. En particulier, les mauvaises oscillations de la reconstruction centrée PPM1 sont accentuées, lorsque de (très) petits pas de temps sont utilisés, alors que les reconstructions décentrées LAG3 et LAG5 sont insensibles à la diminution du pas de temps. Nous noterons également que la reconstruction PPM1 se comporte bien lorsque le pas de temps n'est pas trop petit, ce qui est possible pour un schéma semi-Lagrangien.

4.5 Conclusion

Dans ce chapitre, les schémas de volumes finis ont été étudiés et comparés pour l'approximation numérique du système de Vlasov-Poisson. L'objectif principal était de développer des méthodes non splittées pour l'équation de Vlasov. Deux types de méthodes ont été discutées : (i) une méthode de volumes finis inspirée par [33], et (ii) une méthode basée sur les points

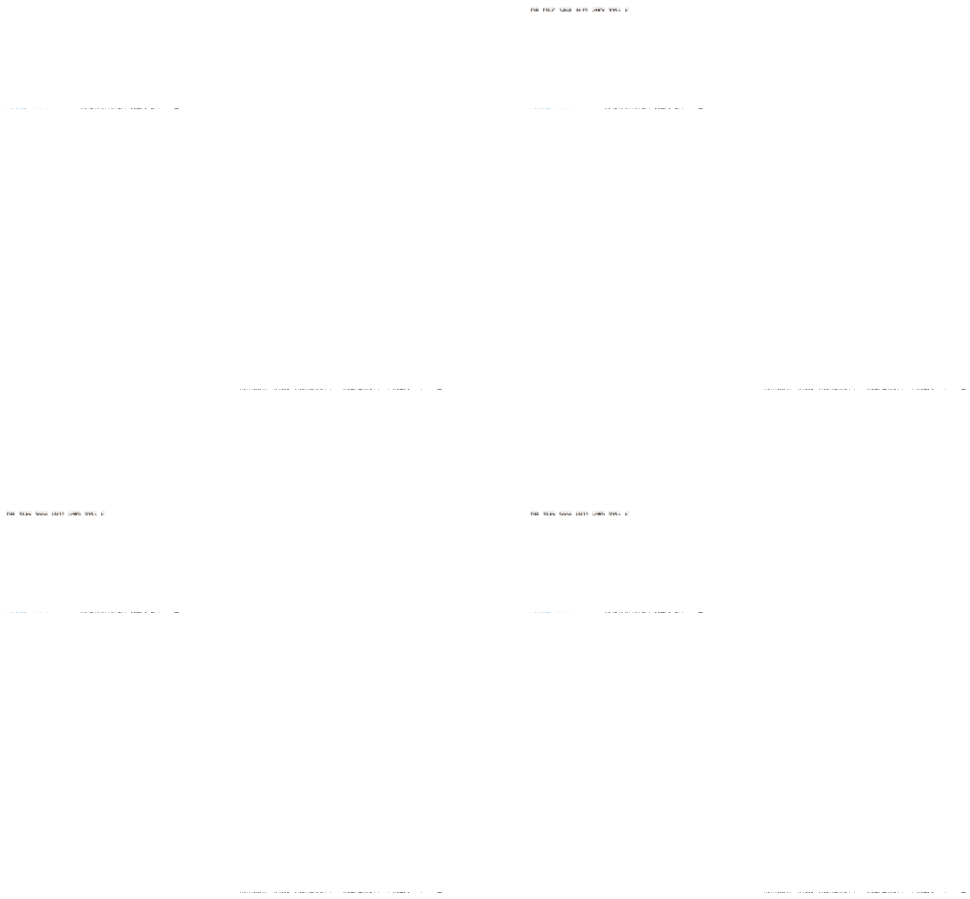


FIGURE 4.8 – Cas test Instabilité double faisceaux : fonction de distribution en fonction de x et v au temps $t = 53$ pour (du haut vers le bas et de gauche à droite) : up5, CD4, Vfsl3-ns, Vfsl5-ns. $N_x = N_v = 128$, $\Delta t = 0.005$.

de Gauss en temps pour l'évaluation des flux. Ces deux méthodes ont un bon comportement dans le contexte des cas-tests académiques des plasmas, par rapport aux méthodes semi-Lagrangiennes standards. De plus, un lien a été effectué entre méthodes de volumes finis et méthodes semi-Lagrangiennes pour l'équation d'advection. En particulier, lorsque le pas de temps Δt tend vers zéro, les méthodes semi-Lagrangiennes récupèrent des caractéristiques des méthodes de volumes finis.

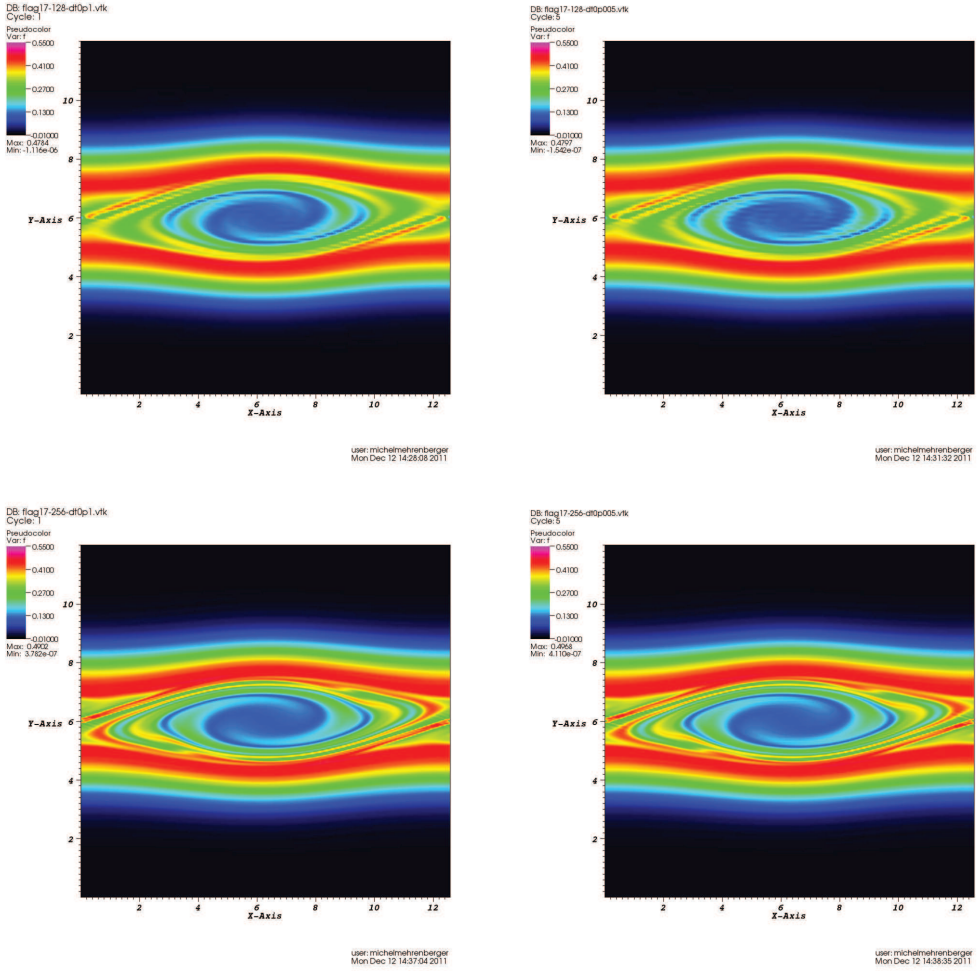


FIGURE 4.9 – Cas test Instabilité double faisceaux : fonction de distribution en fonction de x et v au temps $t = 53$ pour une méthode semi-Lagrangienne avec une reconstruction de Lagrange d'ordre 17 avec $\Delta t = 0.1$ (à gauche), $\Delta t = 0.005$ (à droite), et $N_x = N_v = 128$ (en haut), $N_x = N_v = 256$ (en bas).

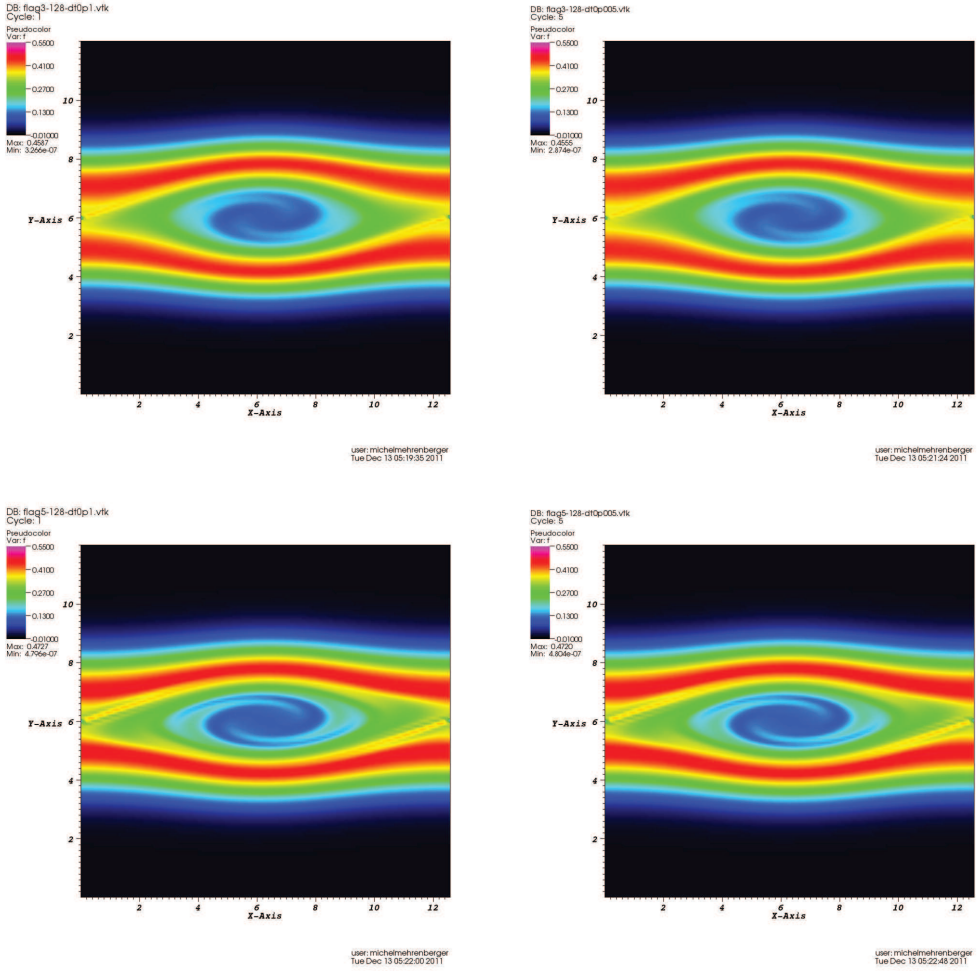


FIGURE 4.10 – Cas test Instabilité double faisceaux : fonction de distribution en fonction de x et v au temps $t = 53$ pour une méthode semi-Lagrangienne avec $N_x = N_v = 128$ et une reconstruction de Lagrange d'ordre 3 (en haut), 5 (en bas) avec $\Delta t = 0.1$ (à gauche), $\Delta t = 0.005$ (à droite).

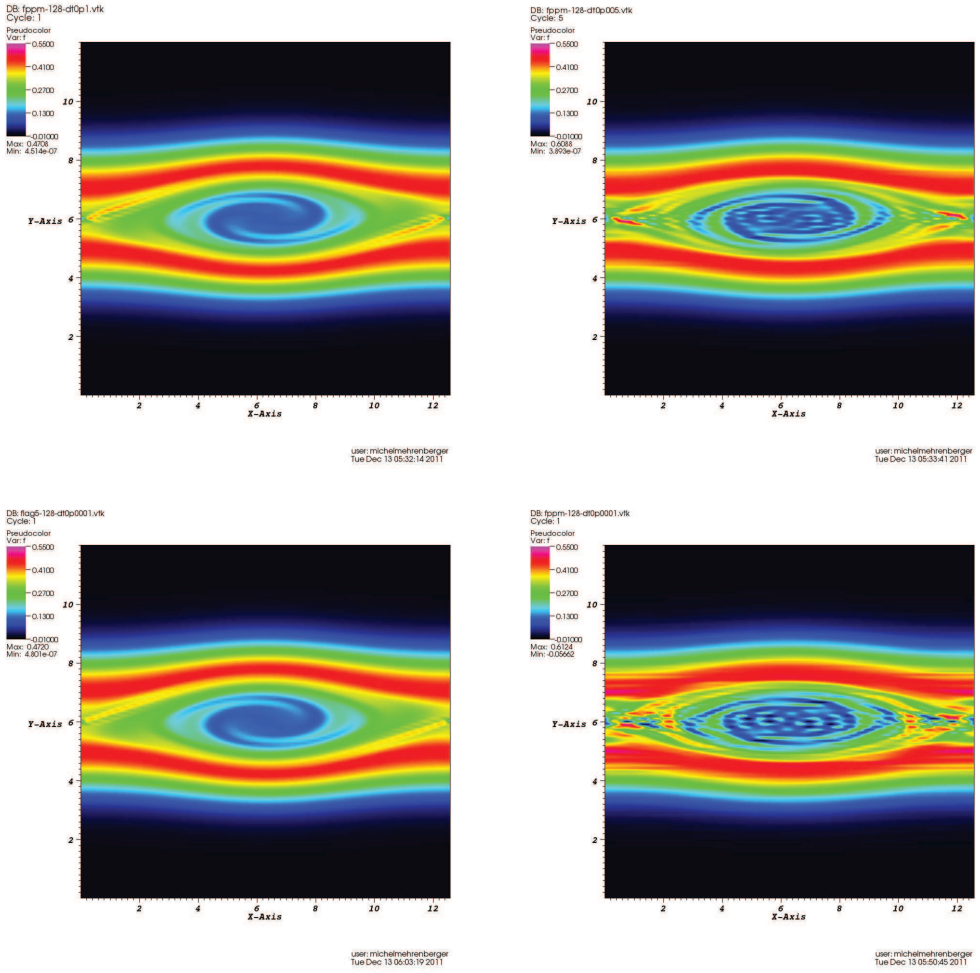


FIGURE 4.11 – Cas test Instabilité double faisceaux : fonction de distribution en fonction de x et v au temps $t = 53$ pour une méthode semi-Lagrangienne avec $N_x = N_v = 128$ et une reconstruction PPM1 avec $\Delta t = 0.1$ (en haut à gauche), $\Delta t = 0.005$ (en haut à droite), $\Delta t = 0.0001$ (en bas à droite), et une reconstruction de Lagrange d'ordre 5 avec $\Delta t = 0.0001$ (en bas à gauche).

Chapitre 5

GPU

L'objectif principal de ce travail est d'utiliser les récentes cartes GPU pour des simulations semi-Lagrangiennes du système de Vlasov-Poisson (3.7.1)-(3.7.2). En effet, il est important de développer de nouveaux algorithmes qui sont hautement scalables dans le champ des simulations de plasma (comme les plasmas de tokamak ou les faisceaux de plasma) pour augmenter leur fiabilité. Les méthodes particulaires ont déjà été testées sur de telles architectures et une bonne scalabilité a été obtenue dans [54, 69].

Nous mentionnons un récent travail précurseur dans la parallélisation sur GPU dans le contexte du code Eulérien gyrocinétique GENE [57]. Les algorithmes semi-Lagrangiens dédiés à la résolution numérique du système de Vlasov-Poisson unidimensionnel ont récemment été implémentés en CUDA (voir [63, 66]). Dans les deux travaux précédents, dans lesquels l'étape d'interpolation est basée sur des splines cubiques, l'efficacité peut atteindre un facteur de 80 dans certains cas. Ici, nous utilisons des algorithmes de complexité plus élevée qui sont basés sur la transformée de Fourier rapide. Nous verrons que nos simulations en GPU vont directement bénéficier de l'importante accélération obtenue par la FFT sur GPU. Ils sont donc aussi très pratiques et nous permettent de tester et de comparer les différents opérateurs d'interpolation (reconstruction par splines ou Lagrangiennes d'ordre très élevé) en utilisant un nombre important de points par direction sur la grille dans l'espace des phases.

Pour réaliser cette tâche, la flexibilité est requise pour passer facilement d'une représentation d'un opérateur à l'autre. Ainsi, les méthodes semi-Lagrangiennes sont reformulées dans un contexte qui permet d'utiliser les routines existantes qui sont optimisées pour la transformée de Fourier rapide. Cette formulation est sous la forme d'une matrice qui possède la propriété de circulance, ce qui est une conséquence des conditions aux bords périodiques. Nous soulignons que de telles conditions aux bords ne sont pas utilisées qu'en x mais également en v ; ceci est rendu possible en considérant comme domaine pour la vitesse $[-v_{\max}, v_{\max}]$, avec $v_{\max}$ assez grand. Notons également que la preuve de la convergence d'un tel schéma numérique peut être obtenu en suivant [52, 23]. Du au fait que de telles matrices sont diagonalisables dans la base de Fourier, le produit matrice-vecteur peut être réalisé efficacement en utilisant la FFT. Dans ce travail, les polynômes de Lagrange de degré impair quelconque $(2d+1)$ et les B-splines de degré quelconque k ont été testés et comparés. Un autre avantage de la formulation en produit matrice-vecteur est que le coût numérique est presque insensible à l'ordre de la méthode. Finalement, puisque les calculs en simple précision sont préférables pour obtenir une performance maximale du GPU, d'autres améliorations ont du être faites par rapport à la méthode semi-Lagrangienne standard. Pour atteindre la précision

nécessaire afin d'observer des phénomènes physiques pertinents, deux modifications ont été apportées : la première est l'utilisation d'une méthode de type δf (voir [63]). La seconde est d'imposer une condition de moyenne nulle en espace sur le champ électrique. Puisque la réponse du plasma est périodique, cette condition est toujours vérifiée.

Le reste de la partie est organisé comme suit. En premier lieu, la reformulation de la méthode semi-Lagrangienne utilisant la FFT est présentée pour le traitement numérique du modèle de Vlasov-Poisson doublement périodique. Ensuite, nous donnerons des détails sur l'implémentation en GPU en mettant en avant les modifications particulières qui furent nécessaires dans le but de dépasser les limitations de la simple précision sur GPU. Finalement, nous présenterons les résultats numériques. Ils contiennent différentes comparaisons entre les différentes méthodes, les ordres d'approximation numérique et leurs performances sur CPU et GPU sur trois cas tests canoniques.

5.1 Implémentation utilisant la FFT

Dans cette partie, nous donnons une formulation explicite des schémas semi-Lagrangiens pour la résolution du système d'équations de Vlasov-Poisson dans le cas d'une double périodicité en utilisant les matrices circulantes. En premier lieu, nous rappellerons le splitting directionnel classique de Strang (voir [1, 49]). Ensuite, le problème est réduit en une succession d'advections constantes unidimensionnelles. Indépendamment de la méthode utilisée ou de l'ordre de l'interpolation, une formulation générale sous forme d'une matrice circulante est proposée pour laquelle l'utilisation de la transformée de Fourier rapide (FFT) est fortement adaptée.

5.1.1 Splitting de Strang

Pour le système d'équations de Vlasov-Poisson (3.7.1)-(3.7.2), il est naturel de séparer le transport dans la direction x du transport dans la direction v. De plus, cela correspond également à un splitting de la partie cinétique et du potentiel électrostatique du Hamiltonien $|v|^2/2 + \phi(t,x)$ où le potentiel électrostatique ϕ est relié au champ électrique à travers l'équation $E(t,x) = -\partial_x \phi(t,x)$.

Pour les simulations de plasma, bien que des splittings d'ordre plus élevés soient possibles (voir [2] et les références contenus dans cet article), le splitting d'ordre 2 de Strang est un bon compromis entre performance et simplicité ce qui explique sa popularité. Il est composé de trois étapes d'advections auxquelles s'ajoutent une mise à jour du champ électrique avant l'advection dans la direction v.

1. Transport en v sur un temps $\Delta t/2$: calcul de $f^\star(x,v) = g(\Delta t/2, x, v)$ en résolvant

$$\partial_t g(t,x,v) + E^n(x)\partial_v g(t,x,v) = 0,$$

   avec la condition initiale $g(0,x,v) = f^n(x,v)$.
2. Transport en x sur un temps Δt : calcul de $f^{\star\star}(x,v) = g(\Delta t, x, v)$ en résolvant

$$\partial_t g(t,x,v) + v\partial_x g(t,x,v) = 0,$$

avec la condition initiale $g(0,x,v) = f^\star(x,v)$.
Mise à jour du champ électrique $E^{n+1}(x)$ en résolvant $\partial_x E^{n+1}(x) = \int f^{\star\star}(x,v)dv - 1$.

3. Transport en v sur un temps $\Delta t/2$: calcul de $f^{n+1}(x,v) = g(\Delta t/2, x, v)$ en résolvant

$$\partial_t g(t,x,v) + E^{n+1}(x)\partial_v g(t,x,v) = 0,$$

avec la condition initiale $g(0,x,v) = f^{\star\star}(x,v)$.

Un des avantages principaux du splitting est que l'algorithme se réduit à une série d'advections unidimensionnelles à coefficients constants. En effet, en considérant le transport le long de la direction x, pour tout v fixé, nous avons affaire à une advection constante. Le même raisonnement reste valable pour la direction v puisque pour tout x fixé, E^n ne dépend pas de la variable advectée v. Nous choisissons de débuter par l'advection en v, ce qui permet d'avoir un multiple entier du nombre de pas de temps pour le champ électrique. La troisième étape de la $n^{ème}$ itération peut être agglomérée avec la $n+1^{ème}$ itération.

5.1.2 Advection constante

Dans cette partie, une reformulation des méthodes semi-Lagrangiennes est proposée dans le cas de l'advection constante avec des conditions périodiques. Soit $u = u(t,x)$ la solution de l'équation suivante pour un certain $c \in \mathbb{R}$:

$$\partial_t u + c\partial_x u = 0, \qquad u(t=0,x) = u_0(x),$$

où l'on considère des conditions périodiques en $x \in [0,L]$. Au niveau continu, la solution vérifie pour tout $t, s \geq 0$ et tout $x \in [0,L]$: $u(t,x) = u(s, x - c(t-s))$. Mentionnons que $x - c(t-s)$ doit être compris *modulo* L à cause des conditions périodiques que nous considérons.
Soit un maillage uniforme de l'intervalle $[0,L]$: $x_i = i\Delta x$ pour $i = 0, \ldots, N$ et $\Delta x = L/N$. Nous introduisons le pas de temps $\Delta t = t^{n+1} - t^n$ pour $n \in \mathbb{N}$. Nous notons que $u_0^n = u_N^n$. En posant

$$u^n = \begin{pmatrix} u_0^n \\ \vdots \\ \vdots \\ u_{N-1}^n \end{pmatrix}, \quad u_i^n \approx u(t_n, x_i), \tag{5.1.1}$$

le schéma semi-Lagrangien s'écrit $u_i^{n+1} = \pi u^n(x_i - c\Delta t)$ où π est une fonction polynomiale par morceaux qui interpole u_i^n pour $i = 0, \ldots, N-1$: $\pi(x_i) = u_i^n$. Ceci peut être reformulé en $u^{n+1} = Au^n$ où A est la matrice définissant l'interpolation. Les conditions périodiques impliquent que la matrice A est circulante :

$$A = \mathcal{C}(a_0, a_1, ..., a_{N-1}) := \begin{pmatrix} a_0 & a_1 & \ldots & \ldots & a_{N-1} \\ a_{N-1} & a_0 & a_1 & \ldots & a_{N-2} \\ \ddots & \ddots & \ddots & \ddots & \ddots \\ \ddots & \ddots & \ddots & \ddots & \ddots \\ a_1 & \ldots & \ldots & a_{N-1} & a_0 \end{pmatrix} \tag{5.1.2}$$

Clairement, la matrice dépend du choix de la reconstruction polynomiale π. Dans la suite, nous donnons certains exemples explicites.

Exemples de différentes méthodes et ordres d'interpolation

Nous avons à évaluer $\pi u^n(x_i - c\Delta t)$. Soit $\beta := -c\Delta t/\Delta x$ le déplacement normalisé qui peut être écrit d'une manière unique sous la forme $\beta = b + b^\star$ avec $(b, b^\star) \in \mathbb{Z} \times [0, 1[$. Cela signifie que le pied de la caractéristique $(x_i - c\Delta t)$ appartient à l'intervalle $[x_{i^\star}, x_{i^\star+1}[$ avec $i^\star + b^\star = i + \beta$, ou $i^\star = i + b$.

1. *Lagrange* 1. Les termes non nuls de la matrice A sont :

$$a_b = 1 - b^\star, \qquad a_{\overline{b+1}} = b^\star.$$

2. *Lagrange* $2d + 1$ (with $2d + 1 \leq N - 1$). Les termes non nuls de la matrice sont :

$$\forall j \in \{-d, \ldots, d+1\}, \quad a_{b+j} = \prod_{k=-d,\ k\neq j}^{d+1} \frac{b^\star - k}{j - k}.$$

3. *B-Spline de degré k.*
   Définissons $B_i^k(x)$ la B-spline de degré k sur le maillage $(x_i)_i$ par la récurrence suivante :

$$B_i^0(x) = \mathbb{1}_{[x_i, x_{i+1}[}(x), \qquad B_i^k(x) = \frac{x - x_i}{k\Delta x} B_i^{k-1}(x) + \left(1 - \frac{x - x_{i+1}}{k\Delta x}\right) B_{i+1}^{k-1}(x).$$

   Ainsi, dans ce cas, la matrice A vaut :

$$A = M \times \mathcal{C}(\underbrace{0, \ldots, 0}_{N-k}, \underbrace{B_0^k(x_1), B_0^k(x_2), \ldots, B_0^k(x_k)})^{-1},$$

   où les termes non nuls de la matrice M sont :

$$\forall j \in \{0, \ldots, k\}, \qquad m_{b-j} = B_0^k(x_{j+b^\star}).$$

Maintenant, en partant de cette reformulation, l'algorithme se réduit à un produit matrice-vecteur à chaque pas de temps. Puisque les matrices sont circulantes, ce produit peut être effectué en passant par la FFT. En effet, les matrices circulantes sont diagonalisables dans l'espace de Fourier [59] tel que :

$$A = UDU^\star,$$

où U est unitaire ($U^\star$ dénote la matrice adjointe de U) et D est diagonale. Ils sont donnés par

$$\begin{aligned} U_{m,k} &= e^{-2i\pi mk/N}, \ \ m, k = 0 \ldots N-1, \\ D_{m,m} &= \sum_{k=0}^{N-1} a_k e^{-2i\pi mk/N}, \ \ m = 0, ..., N-1. \end{aligned}$$

Le produit de U par un vecteur $v \in \mathbb{R}^N$ peut être obtenu en calculant la transformée de Fourier rapide de v. De même, $U^\star v$ peut être obtenu en calculant la transformée inverse de Fourier rapide de v.

Le produit matrice vecteur $Au^n = UDU^\star u^n$ est alors calculé par l'algorithme suivant :

1. Calcul de $U^\star u^n$ en effectuant $\tilde{u} = \mathrm{FFT}^{-1}(u^n)$.
2. Calcul de D en effectuant $\mathrm{FFT}(a)$.
3. Calcul de $w = DU^\star u^n$ en effectuant $D\tilde{u}$.
4. Calcul de Au^n en effectuant $\mathrm{FFT}(w)$.

La complexité de l'algorithme est alors $\mathcal{O}(N \log N)$, indépendamment du degré de la reconstruction polynomiale.

5.2 Implémentation GPU en CUDA

Nous utilisons des noyaux GPU préexistants (NVIDIA) pour la transformée de Fourier rapide, la transposition et le produit scalaire. Notons qu'un tel choix a déjà été effectué dans un contexte plus difficile [57]. Nous aurions préféré utiliser OPENCL (comme dans [56]) afin de ne pas être attachés aux cartes NVIDIA ; mais nous avons eu des difficultés comparativement à la documentation de NVIDIA qui nous a semblé très bien détaillée, en particulier pour la transformée de Fourier rapide.

Les transformées de Fourier rapide sont calculées en utilisant la librairie cufft. Pour la transposition, différents algorithmes sont proposés. La condition $N = N_x = N_v$ est toujours requise pour cette étape. Dans le but de calculer la densité de charge $\rho = \int f(t, x, v)dv$, nous avons adapté la routine `ScalarProd`.

Nous avons également écrit un noyau GPU pour calculer les coefficients de la matrice A. Une formule analytique est utilisée pour chaque coefficient a_i. Dans le cas d'une interpolation de Lagrange de degré $2d + 1$, la complexité change de $\mathcal{O}(Nd)$ à $\mathcal{O}(Nd^2)$ opérations puisque l'algorithme sur CPU que nous avons réécrit était basé sur des différences finies qui ne pouvaient pas être parallélisées.

Les principales étapes de l'algorithme sont :

— Initialisation : la condition initiale est calculée sur CPU et transférée au GPU
— Calcul de la charge de densité initiale ρ sur GPU en utilisant `ScalarProd`
— Transfert de ρ au CPU
— Calcul du champ électrique E sur CPU
— Boucle en temps
  1. Advection sur un temps $\Delta t/2$ en v en utilisant la transformée de Fourier rapide sur GPU
  2. Transposition dans le but de passer dans la direction x sur GPU
  3. Advection sur un temps Δt dans la direction x en utilisant la transformée de Fourier rapide sur GPU
  4. Transposition dans le but de passer dans la direction v sur GPU
  5. Calcul de ρ sur GPU en utilisant `ScalarProd`
  6. Transfert de ρ au CPU
  7. Calcul du champ électrique E sur CPU
  8. Advection sur un temps $\Delta t/2$ en v en utilisant la transformée de Fourier rapide sur GPU

5.3 Questions autour de la simple précision

En principe, les calculs sur GPU peuvent être effectués en utilisant la simple ou la double précision. Cependant, le coût numérique devient très important lorsque l'on choisit la double précision (nous verrons que dans notre cas, le coût est généralement d'un facteur 2) et n'est pas toujours disponible suivant les plateformes. Dans [57] et [66], seule la double précision a été utilisée. Des discussions à propos de la simple précision ont déjà été présentées dans [63]. Dans la suite, nous proposons deux modifications de la méthode semi-Lagrangienne qui permettent l'utilisation de la simple précision en atteignant en même temps la précision du code CPU utilisant la double précision.

5.3.1 La méthode δf

La méthode δf consiste en une séparation d'échelle entre un équilibre et une perturbation telle que la solution soit décomposée comme suit :

$$f(x,v) = \delta f(x,v) + f_{\text{eq}}(v), \quad f_{\text{eq}}(v) = \frac{1}{\sqrt{2\pi}} \exp(-v^2/2).$$

Ainsi, nous nous intéressons à l'évolution en temps de δf qui satisfait

$$\partial_t \delta f + v\partial_x \delta f + E\partial_v[f_{\text{eq}} + \delta f] = 0.$$

Le splitting de Strang présenté dans la partie 5.1.1 est modifié puisque nous advectons δf à la place de f. Puisque f_{eq} ne dépend que de v, les advections en x ne sont pas modifiées. Nous pouvons réécrire l'advection en v comme

$$\partial_t[f_{\text{eq}} + \delta f] + E^\star \partial_v[f_{\text{eq}} + \delta f] = 0,$$

avec la condition initiale $f_{\text{eq}} + \delta f^\star$. Cela signifie que $(f_{\text{eq}} + \delta f)$ est préservé le long des caractéristiques $(f_{\text{eq}} + f^{\star\star})(x,v) = (f_{\text{eq}} + f^\star)(x, v - \Delta t E^\star(x))$. Nous en déduisons que

$$\delta f^{\star\star}(x,v) = \delta f^\star(x, v - \Delta t E^\star(x)) + f_{\text{eq}}(v - \Delta t E^\star(x)) - f_{\text{eq}}(v).$$

ce qui donne la mise à jour de δf pour l'advection en v. Notons que $f_{\text{eq}}(v - \Delta t E^\star(x))$ est une évaluation et non une interpolation.

5.3.2 La condition de moyenne nulle

Le champ électrique est calculé à partir de (3.7.2). Notons que le membre de droite de (3.7.2) est de moyenne nulle, et le champ électrique résultant est aussi de moyenne nulle. Ceci est vrai au niveau continu ; cependant lorsque nous utilisons la simple précision, une erreur cumulative apparaît. Afin d'éviter ce phénomène, nous pouvons imposer numériquement la condition de moyenne nulle sur la grille discrète : à partir de $\rho_k^n \simeq \rho(t^n, x_k) = \int_{\mathbb{R}} f(t^n, x_k, v)dv$, $k = 0, \ldots, N-1$, nous calculons la moyenne :

$$M = \frac{1}{N} \sum_{k=0}^{N-1} \rho_k^n,$$

qui est soustraite à la valeur de ρ_k^n :

$$\tilde{\rho}_k^n = \rho_k^n - M,\ k = 0, \ldots, N-1,$$

d'où $\tilde{\rho}_k^n \simeq \rho(t^n, x_k) - 1$ est numériquement de moyenne nulle. Sans cette modification, nous aurions $\tilde{\rho}_k^n = \rho_k^n - 1$. Nous répétons cette procédure à chaque fois que le champ électrique est calculé : pour un champ électrique calculé numériquement $\tilde{E}_k^n,\ k = 0, \ldots, N-1$, qui peut être de moyenne non nulle, nous calculons $\tilde{M} = \frac{1}{N}\sum_{k=0}^{N-1} \tilde{E}_k^n$, et posons

$$E_k^n = \tilde{E}_k^n - \tilde{M},\ k = 0, \ldots, N-1.$$

Pour calculer le champ électrique, nous utilisons la formule du trapèze :

$$\tilde{E}_{k+1}^n = \tilde{E}_k^n + \Delta x \frac{\tilde{\rho}_k^n + \tilde{\rho}_{k+1}^n}{2},\ k = 0, \ldots, N-1,\ \text{avec } \tilde{E}_0^n \text{ supposé nul de manière arbitraire,}$$

ou en Fourier (avec la FFT). Notons que dans le cas de Fourier, la valeur nulle est automatiquement satisfaite de manière numérique, puisque le mode 0 qui représente la moyenne est supposé égal à 0.
Nous verrons que la condition de moyenne nulle est d'une grande importance pour les résultats numériques. Elle doit être satisfaite avec une assez grande précision. Ceci peut être vu comme relié à un "problème d'annulation" observé dans les simulations PIC [61]. Notons aussi que lorsque nous utilisons la méthode δf, qui est généralement de faible amplitude, une meilleure résolution de la condition de moyenne nulle est atteinte.

5.4 Résultats numériques

Cette section est dévouée à la présentation des résultats numériques obtenus par les méthodes suivantes : la méthode semi-Lagrangienne standard (avec différents opérateurs d'interpolation), en incluant les modifications de la méthode δf et de la condition de moyenne nulle. Les comparaisons entre les simulations en CPU et GPU et les discussions à propos des performances seront données sur trois cas test : Landau damping, instabilité Bump On Tail et les ondes KEEN. Comme opérateur d'interpolation, nous utiliserons par défaut LAG17, l'interpolation de Lagrange d'ordre $2d+1$ avec $d = 8$. De manière similaire, LAG3 signifie $d = 1$ et LAG9 signifie $d = 4$. Nous montrerons également des simulations avec des splines cubiques standards (à but de comparaison), ce qui correspond aux B-splines de degré k avec $k = 4$. Nous utiliserons plusieurs machines pour le GPU : MacBook, irma-gpu1 et hpc. Voir la partie 5.4.4 pour les détails.

5.4.1 Landau Damping

Pour ce premier cas test standard [62], la condition initiale choisie est :

$$f_0(x,v) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{v^2}{2}\right)(1 + \alpha\cos(0.5x)), \quad (x,v) \in [0, 4\pi] \times [-v_{\max}, v_{\max}],$$

avec $\alpha = 10^{-2}$. Nous nous intéressons à l'évolution en temps du champ électrique $\mathcal{E}_e(t) = (1/2)\|E(t)\|_{L^2}^2$ qui est connue comme étant exponentiellement décroissante avec un taux de

$\gamma = 0.1533$ (see [5]). Du au fait que l'énergie électrique décroisse en temps, ce test accentue la différence entre les simulations en simple et double précision.

Les résultats numériques sont donnés dans la Figure 5.1 (haut et milieu gauche). Nous utilisons LAG17, $N = 2048$, $v_{\max} = 8$ comme valeurs par défaut.

Dans le cas de la simple précision (en haut à gauche), nous voyons le bénéfice de l'utilisation de la modification de moyenne nulle (plots 6 et 7 : efft nodelta et trap moyenne nulle nodelta) : les deux résultats sont similaires (nous utilisons la formule du trapèze pour le champ électrique ou Fourier et nous rappelons que dans les 2 cas, la condition de moyenne nulle est satisfaite) et améliorée par rapport aux cas où la condition de moyenne nulle n'est pas imposée dans le cas trapézoïdal (plot 8 : trap nodalta). 23 oscillations correctes sont atteintes jusqu'au temps $t = 50$ pour les plots 6 et 7 (les deux dernières oscillations sont cependant moins bien décrites), tandis que nous avons seulement 16 oscillations correctes jusqu'au temps $t = 34.8$ pour le plot 8 avant la saturation. Si nous utilisons la méthode δf, nous observons d'autres améliorations (plots 1 à 5) : nous gagnons 4 oscillations (donc nous avons 27 oscillations au total) jusqu'au temps $t = 60$, et le champ électrique est sous $6 \cdot 10^{-6} < e^{-12}$. Notons que dans le cas où nous utilisons la méthode δf, ajouter la modification de moyenne nulle n'a pas d'impact ici ; d'un autre côté, les résultats avec la méthode δf sont meilleurs que les résultats avec la modification de moyenne nulle sur ce graphique. Nous avons également ajouté un résultat sur une machine plus ancienne (plot 9 : MacBook), qui conduit à de médiocres résultats (juste 9 oscillations jusqu'au temps $t = 19$ pour la pire méthode). Ainsi les résultats, qui ne sont pas donnés ici, sont différents en appliquant les modifications ; par exemple, nous obtenons 28 oscillations correctes en utilisant la méthode δf avec modification de moyenne nulle. Le standard en point flottant ne doit pas être satisfait ici ce qui pourrait expliquer la différence dans les résultats.

Dans le cas de la double précision (en haut à droite), nous pouvons obtenir des résultats plus précis. En utilisant la méthode δf ou la modification de moyenne nulle (la différence entre les deux options est moins visible), nous obtenons 92 oscillations correctes jusqu'au temps $t = 206$, le champ électrique passe sous $6 \cdot 10^{-13} < e^{-28}$, et nous supposons que nous pourrions ajouter 11 oscillations de plus jusqu'au temps $t = 231$ (nous voyons que les effets de la taille de la grille polluent les résultats), pour obtenir 103 oscillations avec un champ électrique sous $6 \cdot 10^{-14} < e^{-30}$, mais nous sommes limités ici en double précision à $N = 2048$. Une simulation en CPU avec $N = 4096$ confirme les résultats. Nous voyons également les effets de grille (runs avec $N = 1024$) et la vitesse (runs avec $v_{\max} = 6$). Notons que le plot 6 (trap nodelta 1024 v6) donne de moins bons résultats comparé aux autres plots : la taille de la grille est trop petite ($N = 1024$), le domaine en vitesse aussi ($v_{\max} = 6$) et surtout il n'y a pas de modification de moyenne nulle ou de méthode δf. Dans ce cas, nous atteignons uniquement le temps $t = 100$. Nous référons à [70, 58] pour d'autres résultats numériques et discussions et au célèbre travail [65] en ce qui concerne les résultats théoriques. Dans [70], il est également mentionné que nous devons prendre un domaine en vitesse assez grand et prendre assez de points sur la grille. Concernant le GPU et la simple précision, le bénéfice de la méthode δf a déjà été traité dans [63] : 29 oscillations correctes étaient obtenues dans le cas de la simple précision avec la modification δf, 13 oscillations correctes sans la modification et le temps $t = 100$ était atteint dans le cas CPU (N était fixé à 1024 et $v_{\max}$ à 6).

Sur la Figure 5.1 au milieu à gauche, nous affichons l'erreur de masse qui est calculée comme $|\hat{\rho}_0 - 1|$. Nous voyons clairement l'impact entre la conservation de la masse et les

résultats précédents. Nous pouvons également noter que la modification de moyenne nulle n'améliore pas vraiment la conservation de la masse (juste une faible amélioration à la fin, plots 2, 3, 4), mais a un effet bénéfique sur le champ électrique : le mauvais comportement de la conservation de masse n'est pas propagé au champ électrique. D'un autre côté, la méthode δf améliore clairement la conservation de la masse. Nous voyons également l'effet de prendre un domaine en vitesse trop petit, dans le cas de la double précision.

5.4.2 Bump on tail

Pour ce second cas test standard, la condition initiale est considérée comme une perturbation spatiale et périodique de deux Maxwelliennes (voir [49])

$$f_0(x,v) = \left(\frac{9}{10\sqrt{2\pi}}\exp\left(-\frac{v^2}{2}\right) + \frac{2}{10\sqrt{2\pi}}\exp(-2(v-4.5)^2)\right)(1+0.03\cos(0.3x))$$

où $(x,v) \in [0,20\pi] \times [-9,9]$. Le modèle de Vlasov-Poisson préserve des quantités physiques avec le temps, appelées fonctions de Casimir, qui seront utilisées pour comparer les différentes implémentations. Particulièrement, nous regardons l'historique en temps de l'énergie $\mathcal{E}$ du système, qui est la somme de l'énergie cinétique $\mathcal{E}_k$ et de l'énergie électrique $\mathcal{E}_e$:

$$\mathcal{E}(t) = \mathcal{E}_k(t) + \mathcal{E}_e(t) = \int_0^{4\pi}\int_{\mathbb{R}} f(t,x,v)\frac{v^2}{2}dvdx + \frac{1}{2}\int_0^{4\pi} E^2(t,x)dx.$$

Comme dans le cas précédent, l'évolution en temps de l'énergie électrique est choisie comme diagnostic.

Les résultats sont donnés dans la Figure 5.1 (au milieu à droite et en bas) et sur la Figure 5.2.

Nous voyons sur la Figure 5.1 au milieu à droite l'évolution en temps du champ électrique. Les résultats en simple et double précision sont comparés. Dans le cas de la simple précision, la méthode δf avec calcul par FFT du champ électrique (plot 3 : simple précision et delta) est le gagnant et la méthode basique sans modifications avec calcul du champ électrique en utilisant la formule du trapèze (plot 7 : trap single no delta) conduit au moins bon résultat. Les calculs en double précision conduisent à de meilleurs résultats et les différences sont faibles : les plots 1 (double delta) et 2 (double non delta) sont indistinguables et le plot 8 (trap double no delta) est seulement différent à la fin. Ainsi, de telles modifications ne sont pas si essentielles dans le cas de la double précision. Nous observons ensuite pour les mêmes runs, l'évolution de l'erreur de masse (en bas à gauche) et le premier mode de ρ en valeur absolue (en bas à droite). Nous notons que l'erreur de masse s'accumule linéairement en temps. Ici, il n'y a pas d'erreur provenant du domaine en vitesse, car $v_{\max}$ est assez grand ($v_{\max} = 9$ dans tous les runs). L'évolution du premier mode de ρ est très instructif : on voit qu'il se développe de façon exponentielle des erreurs d'arrondi et les différents runs mènent à des résultats tout à fait différents. La perte de masse peut devenir critique dans le cas de la simple précision (pas de réel impact dans le cas de la double précision n'est détecté) et des implémentations sans accumulation d'erreur de masse seraient désirables. La méthode δf améliore les résultats, mais l'erreur de masse s'accumule encore et plus que dans le cas de la double précision.

Sur la Figure 5.2, nous voyons les mêmes diagnostics dans le cas de la double précision. Nous faisons varier le nombre de points sur la grille, le degré d'interpolation et le pas de

temps. En prenant des pas de temps plus petits, nous pouvons augmenter le temps avant la fusion de deux tourbillons parmi trois qui conduit à une répartition du champ électrique. Des interpolations de degrés plus élevés conduisent à de meilleurs résultats (dans le sens où la répartition apparaît plus tard), pour des résolutions de grilles pas trop élevées. Lorsque $N = 2048$, des interpolations d'ordre plus bas semblent être meilleures, puisqu'elles introduisent plus de diffusion, alors que les schémas d'ordre élevés essayent de capturer les petites échelles, qui sont plus difficiles à traiter en temps long. Des méthodes adaptatives et des méthodes avec de faibles erreurs d'arrondis en simple précision pourraient être utiles pour obtenir de meilleurs résultats.

5.4.3 Ondes KEEN

Dans ce dernier et plus complexe cas test, à la place de considérer une perturbation de la condition initiale, nous ajoutons un champ électrique externe de guidage E_{app} aux équations de Vlasov-Poisson :

$$\partial_t f + v\partial_x f + (E - E_{\text{app}})\partial_v f = 0,\ \partial_x E = \int_{\mathbb{R}} f dv - 1,$$

où $E_{\text{app}}(t,x)$ est de la forme : $E_{\text{app}}(t,x) = E_{\max} k a(t) \sin(kx - \omega t)$, où

$$a(t) = \frac{0.5(\tanh(\frac{t-t_L}{t_{wL}}) - \tanh(\frac{t-t_R}{t_{wR}})) - \epsilon}{1-\epsilon},\ \epsilon = 0.5\left(\tanh\left(\frac{t_0 - t_L}{t_{wL}}\right) - \tanh\left(\frac{t_0 - t_R}{t_{wR}}\right)\right)$$

est l'amplitude, $t_0 = 0$, $t_L = 69$, $t_R = 307$, $t_{wL} = t_{wR} = 20$, $k = 0.26$, $\omega = 0.37$ et $E_{\max} = 0.2$. La condition initiale vaut

$$f_0(x,v) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{v^2}{2}\right), \quad (x,v) \in [0, 2\pi/k] \times [-6, 6].$$

Voir [51, 68] pour des détails sur ce cas test physique. Son importance provient du fait que les ondes KEEN représentent de nouvelles oscillations mutimodes non stationnaires de plasmas cinétiques non linéaires sans limite fluide et sans limite linéaire. Ils sont des états de l'auto-organisation du plasma qui ne ressemblent pas à la manière dont les ondes sont initiées. A faible amplitude, ils ne sont pas capables de se former. Les ondes KEEN ne peuvent pas exister hors des courbes de dispersion de petites vagues d'amplitude classique, sans qu'une structure tourbillonnaire autonome soit créée dans l'espace des phases, et suffisamment de particules y soient pris au piège pour maintenir le champ auto-cohérent, longtemps après que le champ de guidage ait été désactivé. Voir [55] pour une méthode numérique alternative de simulation du système de Vlasov-Poisson utilisant l'approximation de Galerkin Discontinu dans le cas test des ondes KEEN.
Comme diagnostics, nous considérons ici différentes captures d'écran de $f - f_0$ à des temps différents : $t = 200, t = 300, t = 400, t = 600$ et $t = 1000$.

Nous considérons d'abord le temps $t = 200$ (en haut à gauche dans la Figure 5.3). A ce temps, toutes les captures d'écran sont identiques donc nous n'en présentons qu'une (GPU en simple précision et une grille de 1024^2 points). Les cinq autres graphiques de cette figure sont pris au temps $t = 300$. Nous voyons qu'à ce temps, il y a encore convergence car le

graphique au milieu à droite (GPU simple précision, $N = 4096$ et $\Delta t = 0.1$) est identique à celui qui est en bas à gauche (GPU en simple précision, $N = 4096$ et $\Delta t = 0.01$).
La Figure 5.4 présente différentes captures d'écran au temps $t = 400$ et $t = 600$. Au temps $t = 400$, le graphique en haut à gauche (GPU en simple précision, $N = 2048$, $\Delta t = 0.1$) est similaire à celui en haut à droite (CPU, $N = 2048$, $\Delta t = 0.05$), ce qui montre que les codes en CPU et en GPU donnent les mêmes résultats. Avec 4096 points (au milieu à gauche), nous observons une petite différence avec le cas 2048 points. Entre les captures au temps $t = 400$ et ceux au temps $t = 600$, nous observons l'émergence de diffusion.

Le temps $t = 1000$ est considéré sur la Figure 5.5. Nous voyons qu'il n'y a plus convergence à ce temps : il y a un décalage, mais la structure reste la même. Nous comparons également différents interpolateurs (splines cubliques, LAG 3, LAG 9, LAG 17). Lorsque l'ordre de l'interpolation est élevé (graphique en haut à droite : CPU, LAG 17, $\Delta t = 0.05, N_x = 512, N_v = 4096$) il y a apparition de fines structures. A ce temps, nous observons de petites différences entre les résultats en CPU (graphique au milieu à droite : CPU, LAG 3, $\Delta t = 0.05, N = 4096$) et les résultats en GPU (graphique en bas à gauche : GPU, LAG 3, $\Delta t = 0.05, N = 4096$), mais il n'y a pas de décalage.

La Figure 6.3 (au temps $t = 1000$) montre la différence entre la simple et la double précision quand la valeur de N change. Les deux graphiques du haut montrent le cas $N = 1024$, celui de gauche est en simple précision tandis que celui de droite est en double précision. Nous voyons qu'il y a très peu de différences. Lorsque $N = 2048$, les résultats sont différents en simple précision (graphique au milieu à gauche) et en double précision (graphique au milieu à droite). Lorsque $N = 4096$, le code ne fonctionne pas en double précision donc nous comparons les résultats en simple précision en GPU avec $\Delta t = 0.05$ (graphique en bas à gauche) et $\Delta t = 0.01$ (graphique en bas à droite). Il y a quelques différences dues de la non-convergence. De plus, nous voyons qu'il y a plus de filamentation lorsque N augmente.

La figure 5.7 montre l'évolution en temps de la valeur absolue des premiers modes de Fourier de ρ. Nous voyons que la simple précision peut modifier les résultats en temps long (en haut à gauche). Le code GPU est validé en double précision (en haut à droite). Nous observons clairement le bénéfice de la méthode δf en GPU simple précision (au milieu à gauche), où il n'a pas d'effet sur le cas de la double précision (au milieu à droite). D'autres plots sont donnés avec $N = 4096$ (en bas à droite et à gauche). Avec de plus petits pas de temps, de petites oscillations apparaissent dans le code GPU en simple précision (en bas à droite). Dans tous les plots, nous ne voyons pas de différences au début ; les différences apparaissent en temps long, comme cela était le cas pour les plots de la fonction de distribution.

5.4.4 Résultats de performance

Caractéristiques. Nous avons testé le code sur différentes machines avec les caractéristiques suivantes :

- GPU
  - (1) = irma-gpu1 : NVIDIA GTX 470 1280 Mo
  - (2) = hpc : GPU NVIDIA TESLA C2070
  - (3) = MacBook : NVIDIA GeForce 9400M
- CPU
  - (4) = MacBook : Intel Core 2 Duo 2.4 GHz
  - (5) = irma-hpc2 : Six-Core AMD Opteron(tm) Processor 8439 SE

— (6) = irma-gpu1 : Intel Pentium Dual Core 2.2 Ghz 2Gb RAM
— (7) = MacBook : Intel Core i5 2.4 GHz

Nous mesurons, dans les codes GPU, la proportion de FFT qui consiste en : transformation $1D$ des données réelles en données complexes, calcul de la FFT, multiplication complexe, calcul de la FFT inverse, transformation en données réelles (dont l'addition dans le cas de la modification δf, si nous utilisons la méthode δf). Nous avons ajouté un diagnostic pour avoir la proportion de temps de la routine `cufftExec` ; notons que ce diagnostic peut modifier très légèrement les mesures de temps (lorsque ceci est le cas ; de nouvelles mesures sont données entre crochets, voir la table 8.1).

Lorsque le nombre de cellules augmente, la proportion de temps de la FFT augmente également, comme le montre la Table 8.1 (cas test des ondes KEEN avec modification δf) ou la Table 8.2 (cas test des ondes KEEN sans modification δf). Notons que le temps d'initialisation et le temps du diagnostic 2D ne sont pas inclus dans le temps total.

Les résultats avec un code CPU (vlaso) sans OpenMP sont donnés en haut de la Table 5.3. Dans ce code, le cas test utilisé est celui du Landau damping avec une advection en x de temps $\Delta t/2$ suivi d'une advection en v de temps Δt et d'une advection en x de temps $\Delta t/2$. La dernière advection en x de l'itération n est fusionnée avec la première advection en x de l'itération $n+1$.

Les résultats avec Selalib (bas de la Table 5.3) sont obtenus avec OpenMP. Nous utilisons 2 processeurs pour (4), 24 processeurs pour (5), 2 processeurs pour (6) et 4 processeurs pour (7).

Dans le but de comparer les performances, nous introduisons le nombre de MA qui représente le nombre de millions de points advectés par seconde : $MA = \frac{N_{step} \times N_{adv} \times N^2}{10^6 \times \text{Temps total}}$ et le nombre d'opérations par seconde (en GigaFLOPS) donné par :

$$GF = \frac{N_{step} \times N_{adv} \times (2N \times 5N\log(N) + 6N^2)}{10^9 \times \text{Temps total}} \quad \text{avec des données complexes (GPU)}$$

$$GF = \frac{N_{step} \times N_{adv} \times (N \times 5N\log(N) + 3N^2)}{10^9 \times \text{Temps total}} \quad \text{avec des données réelles (CPU)}$$

où N_{step} réfère au nombre d'itérations en temps et N_{adv} représente le nombre d'advections faites à chaque pas de temps ($N_{adv} = 3$ en GPU et pour le code Selalib ; $N_{adv} = 2$ dans le code vlaso). A chaque advection, nous calculons N fois (GPU en données complexes) ou $N/2$ fois (CPU en données réelles) :

— Une FFT et une FFT inverse avec approximativement $5N\log(N)$ opérations pour chaque calcul de FFT
— Une multiplication complexe qui requiert $6N$ opérations.

La comparaison entre la Table 8.1 et la Table 8.2 montre que le coût de la méthode δf n'est pas très important sans pour autant être négligeable. Ce coût pourrait être optimisé. Nous profitons clairement de l'énorme accélération des routines FFT sur GPU et donc nous gagnons énormément en choisissant cette approche. La majorité du calcul concerne la FFT, qui est optimisée pour CUDA dans la librairie cufft et qui est transparente pour l'utilisateur. Notons que nous sommes limités ici à $N = 4096$ en simple précision et $N = 2048$ en double précision ; ainsi que nous utilisons la transformation de Fourier complexe ; des transformations réelles optimisées pourraient permettre d'aller encore plus vite. La fusion de deux advections en vitesse pourrait facilement améliorer la vitesse. Des splittings en temps

d'ordre plus élevés pourraient également être utilisés. Ainsi, une meilleure comparaison avec des codes CPU parallélisés peut être envisagée (ici, nous utilisons une implémentation basique en OpenMP qui peut uniquement utiliser 2 processeurs). Nous pouvons également espérer aller sur des grilles plus grandes puisque cufft autorise des tailles de grilles de 128 millions d'éléments en double précision et de 64 millions en simple précision (ici nous utilisons $2^{24} \simeq 16.78 \cdot 10^6$ éléments en double précision ; donc nous pourrions être capables de faire tourner le programme avec $N = 8192$ en simple précision et $N = 4096$ en double précision). Des problèmes de complexité plus élevée (comme les simulations $4D$) vont probablement nécessiter du multi-GPU qui est une autre histoire, voir [57] pour un tel travail.

5.5 Conclusion

Nous avons montré que cette approche fonctionne. La majorité des calculs du code est effectuée par la routine de FFT, qui est optimisée en CUDA dans la librairie cufft, conduisant à d'importants speed-ups et est invisible pour l'utilisateur. Ainsi, la surcharge de temps d'implémentation qui peut être très significative dans d'autres contextes est ici réduite, puisque nous utilisons des routines déjà largement programmées et qui sont déjà optimisées. L'utilisation de la simple précision peut être effectuée sans risque grâce à la méthode δf. Cependant, nous ne sommes pas capables d'obtenir des résultats aussi précis que dans le cas de la double précision. Les cas test que nous avons choisi sont très sensibles aux erreurs de la simple précision. Nous soulignons également que le champ électrique doit satisfaire une condition de moyenne nulle avec assez de précision sur la grille discrète. Pour le moment, nous sommes limités à la même taille en x et en v (nécessaire ici pour l'étape de transposition) et à $N = 2048$ en double précision ($N = 4096$ en simple précision). Nous espérons implémenter une version en quatre dimensions (2 en x et 2 en v) de ce code puis inclure des collisions.

		Simple précision		
	N_x	Time (ms) (**speedup**)	MA	FFT (cufftExec)
(1)	256	703 (**2.8-8.5**)	279.6	0.635 (0.36)
	512	1878 (**4.3-17**)	418.7	0.759 (0.46)
	1024	6229 (**9.6-20**)	505.0	0.841 (0.51)
	2048	21908 (**13-27**)	574.3	0.861 (0.50)
	4096	90093 (**15-52**)	558.6	0.888 (0.54)
(2)	256	1096 [1378] (**1.8-5.5**)	179.3	0.471 [0.59 (0.37)]
	512	2125 [2550] (**3.8-15**)	370.0	0.654 [0.69 (0.48)]
	1024	5684 [6001] (**11-22**)	553.4	0.775 [0.79 (0.59)]
	2048	19871 [20284] (**14-29**)	633.2	0.825 (0.62)
	4096	81943 (**17-57**)	614.2	0.859 (0.66)
(3)	256	5783 (**0.3-1.0**)	33.9	0.773 (0.65)
	512	19936 (**0.4-1.6**)	39.4	0.780 (0.66)
	1024	87685 (**0.68-1.4**)	35.8	0.813 (0.71)

		Double précision		
	N_x	Time (ms) (**speedup**)	MA	FFT (cufftExec)
(1)	256	1304 (**1.5-4.6**)	150.7	0.767 (0.61)
	512	3516 (**2.3-8.8**)	223.6	0.839 (0.67)
	1024	11670 (**5.1-11**)	269.5	0.889 (0.71)
	2048	49925 (**5.7-12**)	252.0	0.916 (0.75)
(2)	256	1653 (**1.2-3.6**)	118.9	0.637 (0.5)
	512	3896 (**2.1-8.0**)	201.8	0.777 (0.66)
	1024	12127 (**4.9-10**)	259.3	0.866 (0.76)
	2048	45753 (**6.3-13**)	275.0	0.897 (0.80)

TABLE 5.1 – Résultats de performance pour le code GPU, nbstep=1000, LAG17, cas test des ondes KEEN avec modification δf : temps total, speedup, MA, proportion FFT/temps total (et temps d'éxécution de cufftExec/temps total).

		Simple précision			
	N_x	Temps (ms) **speedup**	MA	GF	FFT
(1)	256	570 (**3.5-11**)	344.9	29.6	0.573
	512	1421 (**5.6-22**)	553.4	53.1	0.702
	1024	4516 (**13-28**)	696.5	73.8	0.787
	2048	15189 (**19-38**)	828.4	96.0	0.802
	4096	63310 (**22-73**)	795.0	100.1	0.842
(2)	256	1000 (**2.0-6.0**)	196.6	16.9	0.520
	512	2000 (**4.0-15**)	393.2	37.7	0.635
	1024	5067 (**12-25**)	620.8	65.8	0.762
	2048	17692 (**16-33**)	711.2	82.5	0.805
	4096	73488 (**19-63**)	684.8	86.2	0.843
(3)	256	5513 (**0.36-1.1**)	35.6	3.0	0.763
	512	18805 (**0.43-1.6**)	41.8	4.0	0.769
	1024	83312 (**0.72-1.5**)	37.7	4.0	0.804

		Double précision			
	N_x	Temps (ms) **speedup**	MA	GF	FFT
(1)	256	1183 (**1.7-5.1**)	166.1	14.2	0.754
	512	3121 (**2.6-10**)	251.9	24.1	0.826
	1024	10221 (**5.9-12**)	307.7	32.6	0.876
	2048	44244 (**6.5-13**)	284.3	32.9	0.906
(2)	256	1569 (**1.3-3.8**)	125.3	10.7	0.657
	512	3750 (**2.1-8.3**)	209.7	20.1	0.782
	1024	11749 (**5.1-11**)	267.7	28.3	0.865
	2048	44446 (**6.5-13**)	283.1	32.8	0.895

TABLE 5.2 – Résultats de performance pour le GPU, nbstep=1000, LAG17, cas test des ondes KEEN sans modification δf : temps total, speedup, MA, GFlops et proportion FFT/temps total.

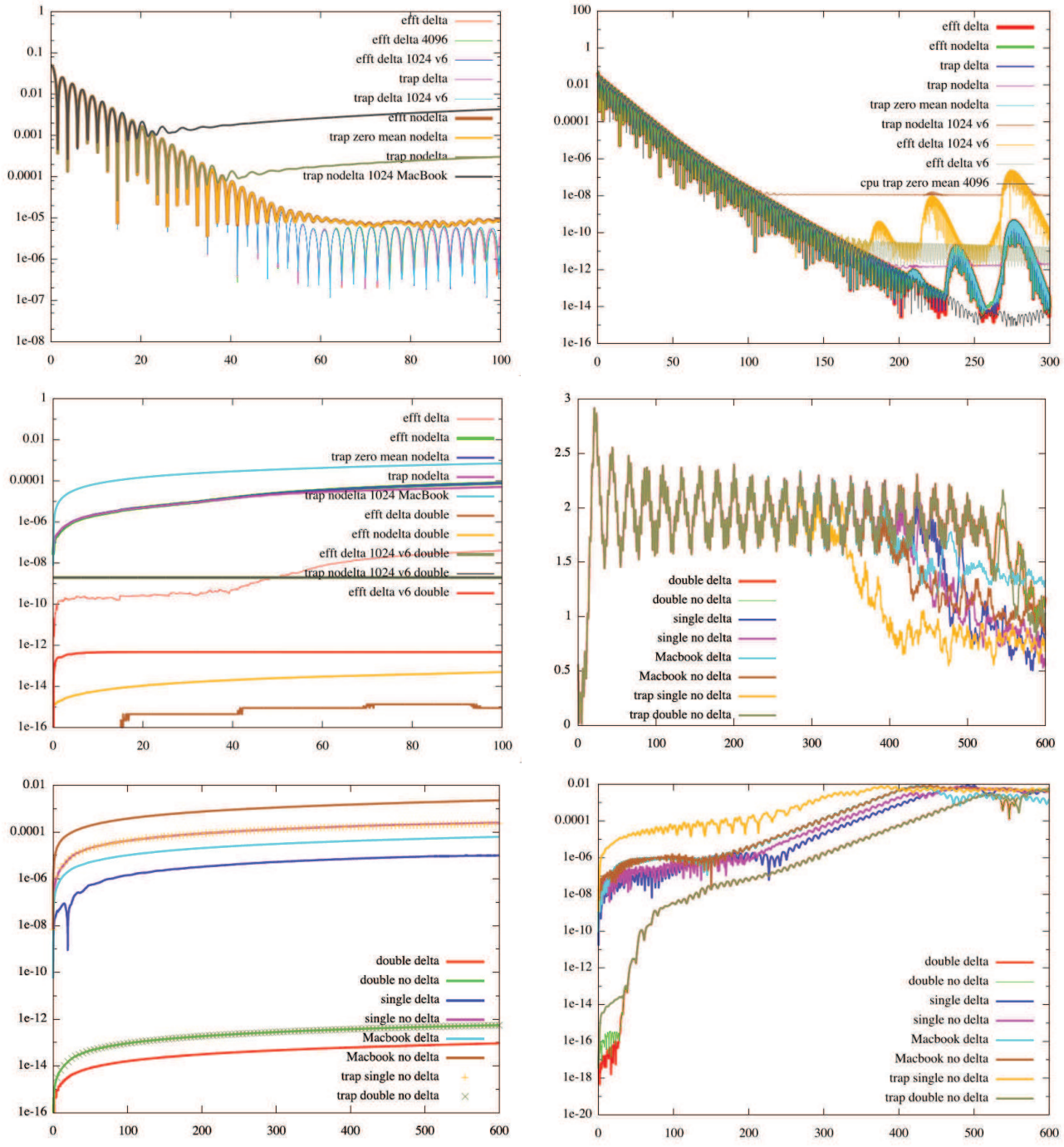


FIGURE 5.1 – Landau damping linéaire. $N = 2048$, $\Delta t = 0.1$, $v_{\max} = 8$, LAG17, irma-gpu1 sur GPU par défaut. Evolution en temps de l'énergie électrique en simple/double précision (en haut à gauche/droite). Erreur de masse $|\hat{\rho}_0 - 1|$ avec la simple précision par défaut (au milieu à gauche). Cas test Bump on tail. $N = 1024, \Delta t = 0.05$, LAG9, irma-gpu1 sur GPU par défaut. Evolution en temps de l'énergie électrique / erreur de masse / premier mode de Fourier de ρ, $|\hat{\rho}_1|$ (milieu à droite / en bas à gauche / en bas à droite). [pour les détails, voir les légendes. efft : champ électrique calculé avec la FFT ; delta = méthode δf ; no delta = sans la méthode δf ; simple = simple précision ; double = double précision ; trap = champ électrique calculé avec la formule du trapèze ; zero mean = modification de la moyenne nulle pour le champ électrique ; cpu = code utilisant un CPU ; 1024 : $N = 1024$; 4096 : $N = 4096$; v6 : $v_{\max} = 6$; Macbook : GPU du MacBook est utilisé].

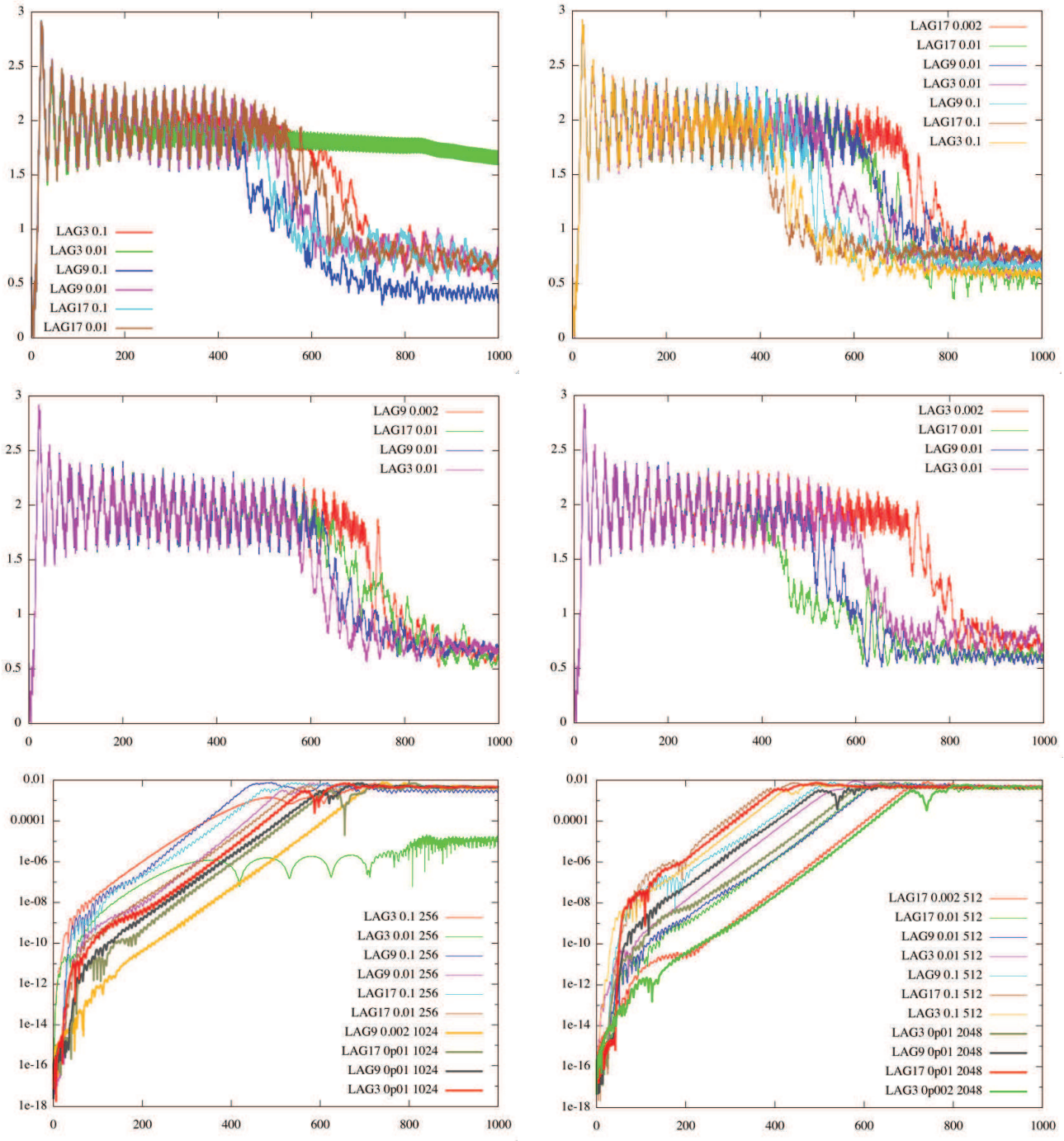


FIGURE 5.2 – Cas test Bump on tail. La double précision est utilisée, irma-gpu1 en GPU. Evolution en temps du champ électrique pour $N = 256, 512, 1024, 2048$ (en haut à gauche, en haut à droite, milieu à gauche, milieu à droite), avec les reconstructions LAG3, LAG9, LAG17 et différents pas de temps $(0.1, 0.01, 0.002)$. Evolution en temps des premiers modes de Fourier , $|\hat{\rho}_1|$ pour $N = 256$ et $N = 1024$ (en bas à gauche), et pour $N = 512$ et $N = 2048$ (en bas à gauche), avec les mêmes reconstructions et pas de temps.

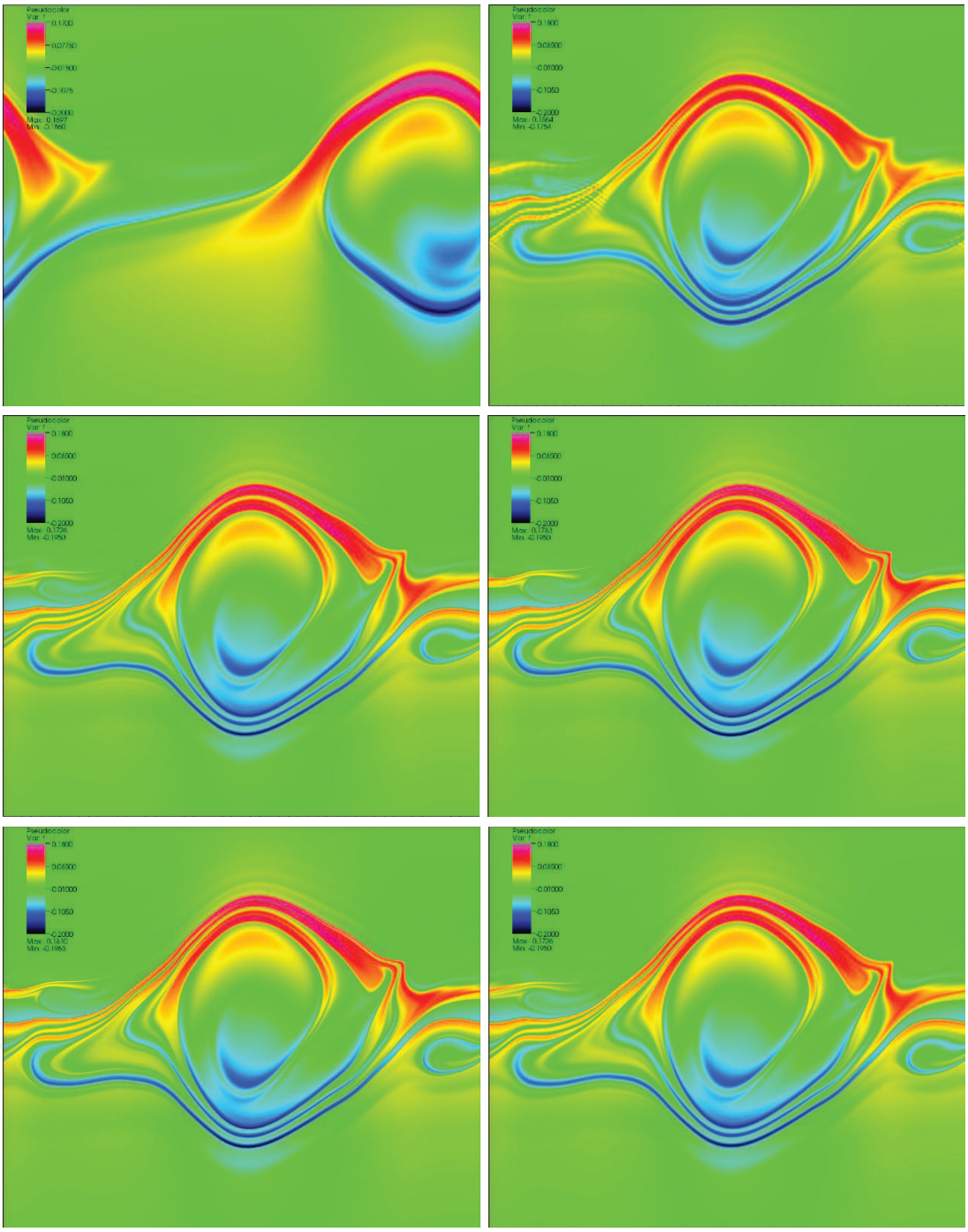


FIGURE 5.3 – Cas test des ondes KEEN (LAG17) : $f(t,x,v) - f_0(x,v)$. Au temps $t = 200$, GPU en simple précision $N = 1024$ (en haut à gauche). Au temps $t = 300$, GPU en simple précision $N = 1024, 2048, 4096$ et $\Delta t = 0.1$ (en haut à droite, milieu gauche, milieu droit). $N = 4096$ et $\Delta t = 0.01$ (en bas à gauche). CPU $N = 2048, \Delta t = 0.1$ (à bas à droite). $(x,v) \in [0, 2\pi/k] \times [0.18, 4.14]$. S'il n'y a pas de modifications d'une figure à l'autre (d'en haut à gauche à en bas à droite), les paramètres ne sont pas répétés.

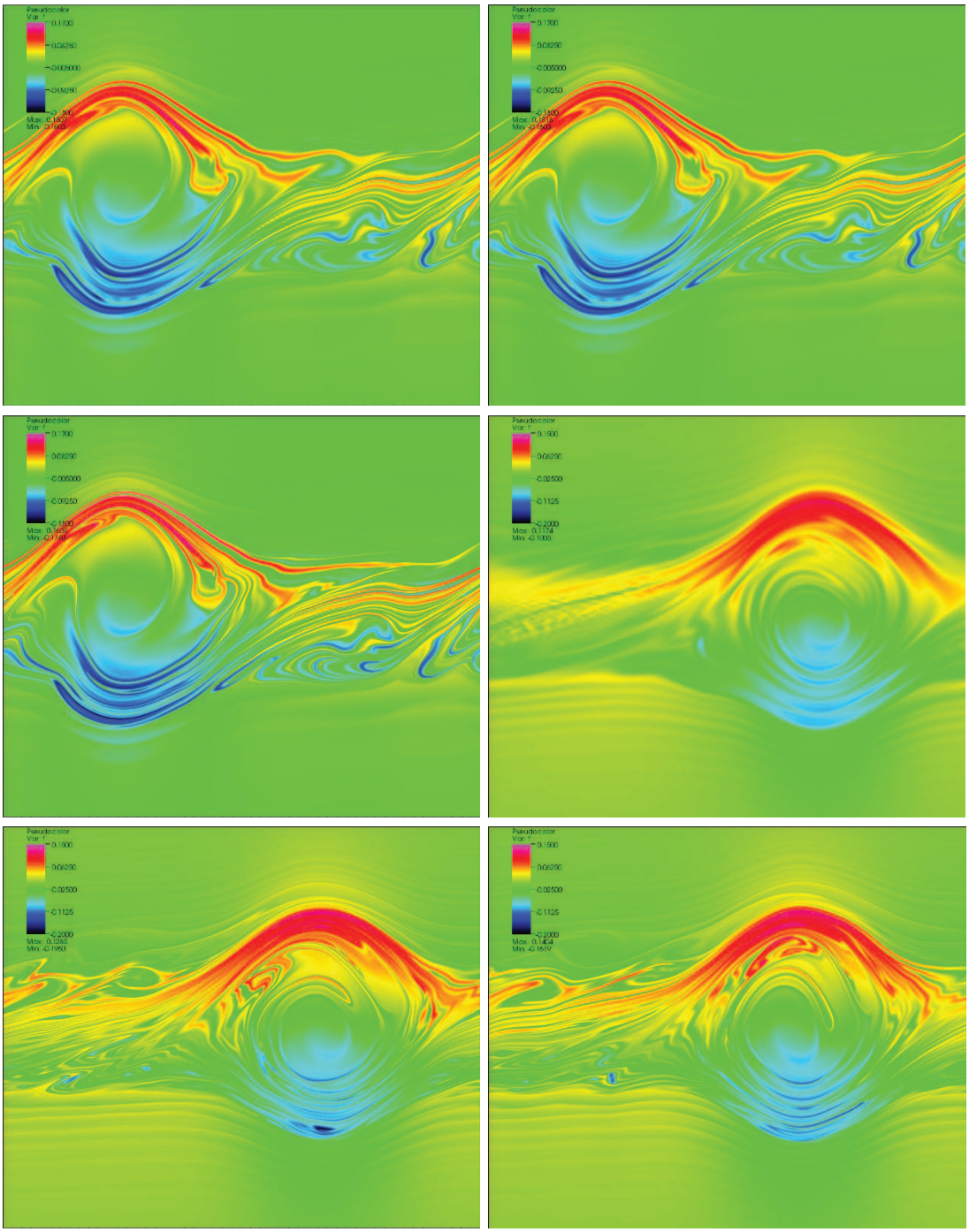


FIGURE 5.4 – Cas test des ondes KEEN (LAG17) : $f(t,x,v) - f_0(x,v)$. Au temps $t = 400$, GPU en simple précision $N = 2048, \Delta t = 0.1$ (en haut à gauche). CPU $\Delta t = 0.05$ (en haut à droite). GPU simple précision $N = 4096, \Delta t = 0.1$, au temps $t = 600$ (milieu à gauche). GPU en simple précision $N = 1024$ (milieu à droite). $\Delta t = 0.01, N = 4096$ (en bas à gauche). CPU $N_x = 512, N_v = 4096$ (en bas à droite). $(x,v) \in [0, 2\pi/k] \times [0.18, 4.14]$. S'il n'y a pas de modifications d'une figure à l'autre (d'en haut à gauche à en bas à droite), les paramètres ne sont pas répétés.

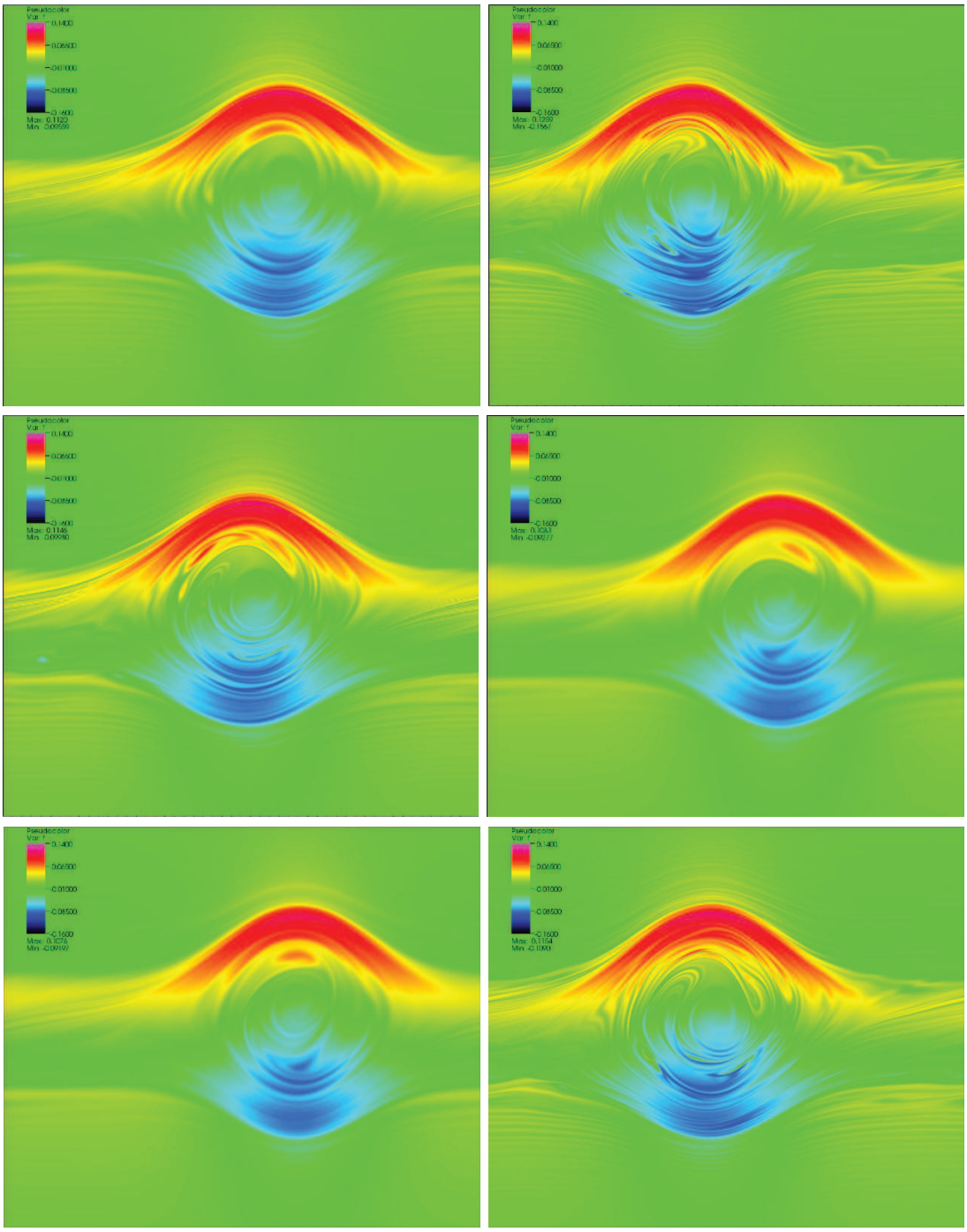


FIGURE 5.5 – Cas test des ondes KEEN : $f(t,x,v) - f_0(x,v)$ au temps $t = 1000$. CPU en splines cubiques, $\Delta t = 0.05, N_x = 512, N_v = 4096$ (en haut à gauche). LAG17 (en haut à droite). $N = 4096$ et les splines cubiques (au milieu à gauche). LAG3 (au milieu à droite). GPU en simple précision (en haut à gauche). LAG9 (en bas à droite). $(x,v) \in [0, 2\pi/k] \times [0.18, 4.14]$. S'il n'y a pas de modifications d'une figure à l'autre (d'en haut à gauche à en bas à droite), les paramètres ne sont pas répétés.

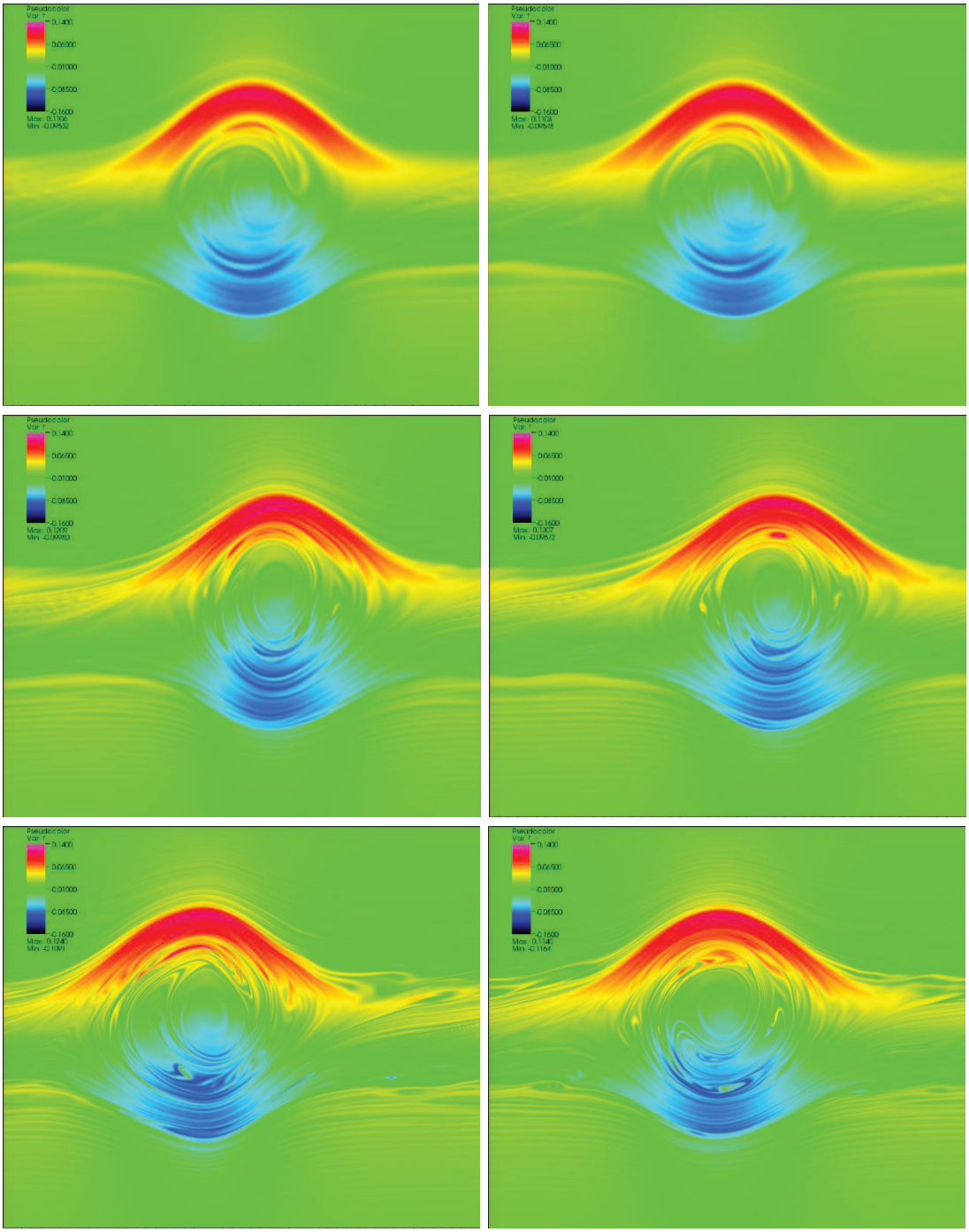


FIGURE 5.6 – Cas test des ondes KEEN (LAG17) : $f(t,x,v) - f_0(x,v)$ au temps $t = 1000$, $\Delta t = 0.05, N = 1024$ par défaut. GPU en simple/double précision (en haut à gauche/droite). $N = 2048$, GPU en simple/double précision (milieu gauche/droite). $N = 4096$, GPU en simple précision (en bas à gauche). $\Delta t = 0.01$ (en bas à droite). $(x,v) \in [0, 2\pi/k] \times [0.18, 4.14]$. S'il n'y a pas de modifications d'une figure à l'autre (d'en haut à gauche à en bas à droite), les paramètres ne sont pas répétés.

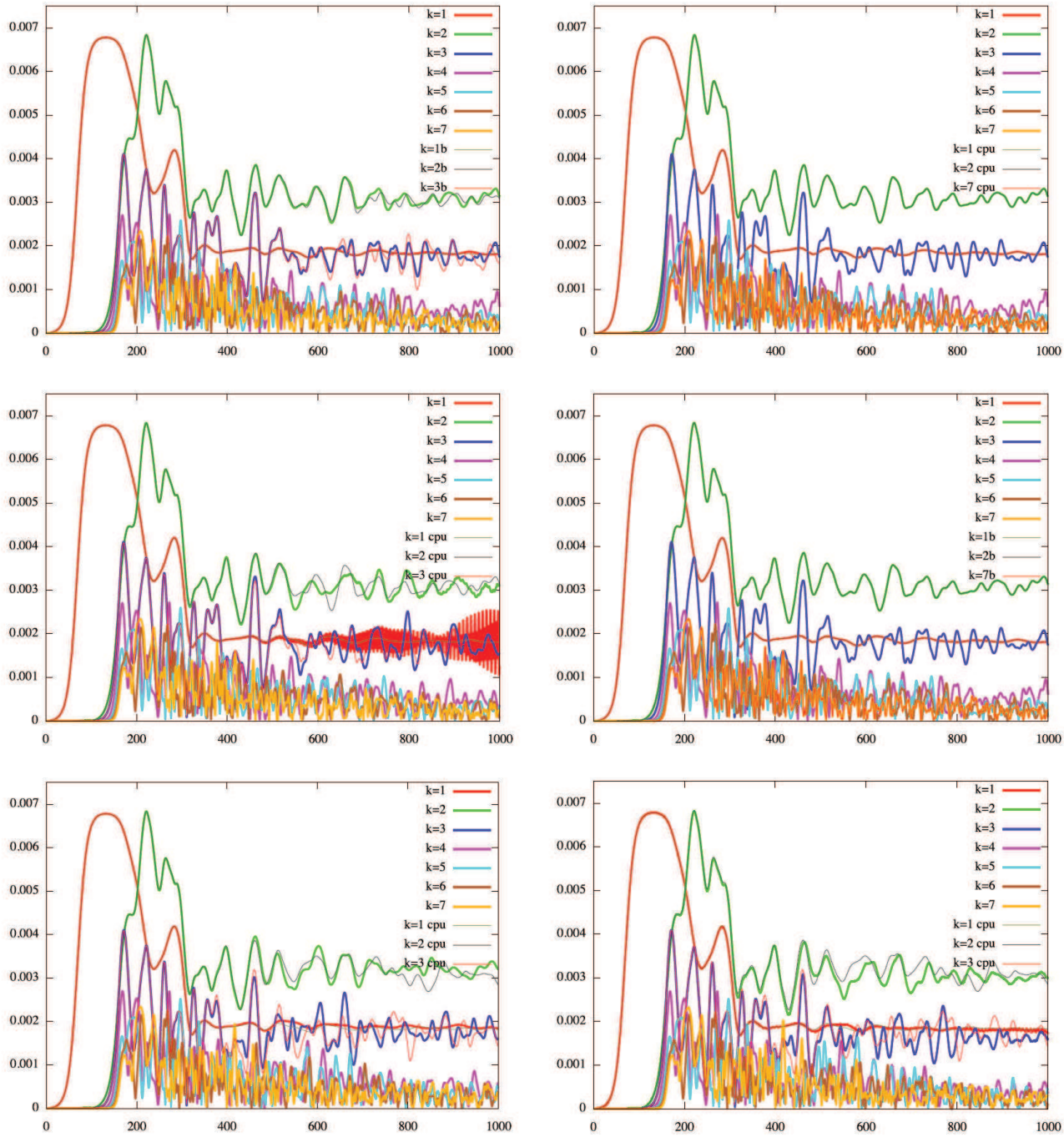


FIGURE 5.7 – Cas test des ondes KEEN (LAG17) : Valeurs absolues des premiers modes de Fourier de ρ (du mode $k = 1$ au mode $k = 7$) vs temps. Méthode δf, avec $N = 2048$ $\Delta t = 0.05$ GPU, double et simple précision (1b,2b,3b) (en haut à gauche). GPU en double précision et CPU en double précision (en haut à droite). Full version GPU en simple précision et CPU avec la méthode δf (milieu à gauche). Full version et version δf, en double précision (milieu à droite). $N = 4096$, GPU et CPU (en bas à gauche). GPU avec $\Delta t = 0.01$ et CPU avec $\Delta t = 0.05$ (en bas à droite). S'il n'y a pas de modifications d'une figure à l'autre (d'en haut à gauche à en bas à droite), les paramètres ne sont pas répétés.

	(4)			(5)		
N_x	Total time	MA	GF	Total time	MA	GF
256	4s	27.4	1.1	4s	28.8	1.2
512	27s	19.2	0.9	18s	28.8	1.3
1024	1min52s	18.7	0.9	2min4s	16.8	0.8
2048	8min16s	16.9	0.9	9min31s	14.6	0.8
4096	41min05s	13.6	0.8	48min16s	11.5	0.7
256	3s	58.0	2.4	4s	43.9	1.8
512	19s	39.6	1.9	8s	90.6	4.3
1024	1min25s	36.8	1.9	1min21s	38.5	2.0
2048	6min41s	31.3	1.8	7min46s	27.0	1.5
4096	34min39s	24.2	1.5	25min33s	32.8	2.0

	(6)			(7)		
N_x	Total time	MA	GF	Total time	MA	GF
256	6s	21.4	0.9	3s	38.8	1.6
512	31s	16.5	0.7	15s	34.7	1.6
1024	2min7s	16.4	0.8	1min18s	26.7	1.4
2048	9min42s	14.4	0.8	5min36s	24.9	1.4
4096	52min20s	10.6	0.6	28min28s	19.6	1.2
256	3s	54.3	2.3	2s	72.6	3.1
512	22s	35.0	1.6	13s	58.7	2.8
1024	1min35s	32.9	1.7	1min0s	52.1	2.7
2048	8min47s	28.3	1.6	4min47s	43.7	2.5
4096	77min31s	10.8	0.6	23min09s	36.2	2.2

TABLE 5.3 – Résultats de performance pour le code vlaso en CPU, nbstep=1000, LAG 17, cas test Landau (haut) : temps total, MA and GFlops. Résultats de performance pour le code CPU Selalib, nbstep=1000, LAG 17, cas test des ondes KEEN sans modification δf (bas) : temps total, MA and GFlops.

Chapitre 6

Schéma SLDG sur maillage non uniforme

Ce chapitre présente la résolution numérique de l'équation de Vlasov-Poisson $1D \times 1D$ en utilisant un schéma Galerkin Discontinu Semi-Lagrangien (SLDG). Un tel schéma a déjà été développé dans [27] et plus récemment dans [25, 28, 30, 74] pour des applications à Vlasov-Maxwell/Poisson.

Un point clé dans de telles applications est d'utiliser un splitting directionnel qui conduit à une succession de problèmes d'advection constante et le schéma a l'avantage de ne pas être restreint à une condition CFL. Dans le cas de l'équation d'advection linéaire à vitesse constante sur un maillage uniforme, le schéma SLDG a une propriété de superconvergence. Un avantage de cette méthode est qu'elle permet aussi de considérer le cas de l'advection constante sur un maillage 1D déstructuré. Nous étudierons le cas d'un maillage non uniforme en vitesse. Nous comparerons ces résultats avec la simulation des ondes KEEN (Kinetic Electrostatic Electron Nonlinear waves), que nous avons introduit dans le chapitre précédent, pour lesquelles nous avons déjà des résultats numériques sur GPU dans le cas d'un maillage uniforme et dont les méthodes ont été adaptées à un maillage non uniforme dans [71]. En effet, le cas d'un maillage uniforme en vitesse ne semble pas être le meilleur choix pour la simulation des ondes KEEN puisque la région de l'espace des phases qui est hautement perturbée est très restreinte en vitesse.

6.1 Cas test des ondes KEEN

Nous résolvons l'équation de Vlasov-Poisson dans le cas test des ondes KEEN :

$$\partial_t f + v\partial_x f + (E - E_{\text{app}})\partial_v f = 0,\ \partial_x E = \int_{\mathbb{R}} f dv - 1, \tag{6.1.1}$$

où le champ électrique de guidage appliqué $E_{\text{app}}(t,x)$ est de la forme

$$E_{\text{app}}(t,x) = E_{\max} k a(t) \sin(kx - \omega t),$$

où

$$a(t) = \frac{0.5(\tanh(\frac{t-t_L}{t_{wL}}) - \tanh(\frac{t-t_R}{t_{wR}})) - \epsilon}{1-\epsilon},$$

$$\epsilon = 0.5\left(\tanh\left(\frac{t_0 - t_L}{t_{wL}}\right) - \tanh\left(\frac{t_0 - t_R}{t_{wR}}\right)\right)$$

est l'amplitude, $t_0 = 0$, $t_L = 69$, $t_R = 307$, $t_{wL} = t_{wR} = 20$, $k = 0.26$, $\omega = 0.37$ et $E_{\max} = 0.2$. La condition initiale vaut

$$f_0(x,v) = \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{v^2}{2}\right), \quad (x,v) \in [0, 2\pi/k] \times [-6, 6].$$

La simulation numérique des ondes KEEN est un challenge puisque ce cas test développe de fines structures en temps long, nécessitant un maillage fin de l'espace des phases ainsi qu'un faible pas de temps.

6.2 Implémentation sur maillage uniforme

Une méthode semi-Lagrangienne classique permettant de résoudre l'équation de Vlasov (6.1.1) est le splitting directionnel. Les techniques de splitting (splitting de Strang ou splittings d'ordre plus élevé) conduisent à

$$\partial_t f + v\partial_x f = 0, \ \partial_t f + (E - E_{\text{app}})\partial_v f = 0,$$

i.e. une succession de $N \in \{N_x, \ N_v\}$ équations d'advection $1D$ à vitesse constante puisque le champ électrique E ainsi que le champ électrique de guidage appliqué E_{app} ne dépendent pas de la vitesse.

Considérons une subdivision du domaine Ω par un maillage uniforme de N cellules $C_i = [x_i, x_{i+1}]$ avec $i = 0, ..., N-1$. Soit Δx le pas d'espace, Δt le pas de temps et $t^n = n\Delta t$. Au niveau continu, la solution d'une équation d'advection constante

$$\partial_t f + a\partial_x f = 0, \ f = f(t,x), \qquad x \in \Omega, \ t \geq 0 \tag{6.2.1}$$

vérifie $f(t^{n+1}, x_i) = f(t^n, x_i - a\Delta t)$. Nous avons déjà détaillé, dans ce manuscrit, plusieurs méthodes de résolution de cette équation sur maillage uniforme que nous résumons dans les deux parties suivantes.

6.2.1 Schémas de Lagrange et splines cubiques

Nous rappelons la méthode utilisée dans le chapitre précédent et pour laquelle une accélération en GPU a été mise en oeuvre en CUDA. Afin de mettre à jour $f_i^{n+1} \simeq f(t_{n+1}, x_i)$, nous utilisons le fait que

$$f_i^{n+1} \simeq f(t^{n+1}, x_i) = f(t^n, x_i - a\Delta t)$$

où le terme $f(t^n, x_i - a\Delta t)$ est calculé par interpolation au temps t^n. Les conditions aux bords périodiques nous permettent d'écrire $f^{n+1} = Af^n$ où A est une matrice circulante de taille $N \times N$:

$$A := \begin{pmatrix} a_0 & a_1 & \dots & \dots & a_{N-1} \\ a_{N-1} & a_0 & a_1 & \dots & a_{N-2} \\ \ddots & \ddots & \ddots & \ddots & \ddots \\ \ddots & \ddots & \ddots & \ddots & \ddots \\ a_1 & \dots & \dots & a_{N-1} & a_0 \end{pmatrix}.$$

Les matrices circulantes étant diagonalisables dans la base de Fourier, l'utilisation de la FFT permet, dans ce cas, un produit matrice/vecteur efficace. De plus, nous utilisons la même forme pour les différentes interpolations :

— **LAG(2d+1)** : Lagrange symétrique d'ordre $p = 2d + 1$
— **SPL(p)** : B-splines d'ordre p
— **SPL3** correspond aux splines cubiques classiques.

La complexité de calcul est alors largement indépendante de p.

6.2.2 Schéma SLDG

Nous rappelons le principe du schéma SLDG sur maillage uniforme décrit dans le chapitre 3 afin de résoudre l'équation d'advection linéaire (6.2.1). Soit $d \in \mathbb{N}$. Sur chaque cellule $C_i = [x_i, x_{i+1}]$, nous plaçons $d + 1$ points de Gauss $\{x_{ij}\}_{(i,j)\in\{0,\dots,N-1\}\times\{0,\dots,d\}}$ et nous considérons φ_{ij} le polynôme de Lagrange aux points de Gauss x_{ij} restreint à la cellule i. En écrivant $f^n \approx f(t^n, \cdot)$ sous la forme

$$f^n(x) = \sum_{i,j} f_{ij}^n \varphi_{ij}(x)$$

les degrés de liberté $f_{ij}^n \approx f(t^n, x_{ij})$ sont donnés par

$$\omega_j \Delta x f_{ij}^n = \int_{\mathbb{R}} f^n(x)\varphi_{ij}(x)dx.$$

En utilisant l'équation d'advection pour mettre à jour les degrés de liberté, nous obtenons le schéma :

$$\omega_j \Delta x f_{ij}^{n+1} = \int_{\mathbb{R}} f^n(x - a\Delta t)\varphi_{ij}(x)dx.$$

Nous définissons $i^\star$ et α tel que $x_i - a\Delta t = x_{i^\star} + \alpha\Delta x$. Nous pouvons alors obtenir une formulation explicite du schéma :

$$\begin{aligned}\omega_j f_{ij}^{n+1} &= \sum_{\ell=0}^{d} f_{i^\star,\ell}^n \int_{\mathbb{R}} \varphi_\ell(\alpha + s)\varphi_j(s)ds \\ &\quad + \sum_{\ell=0}^{d} f_{i^\star+1,\ell}^n \int_{\mathbb{R}} \varphi_\ell(\alpha + s - 1)\varphi_j(s)ds\end{aligned}$$

où φ_ℓ est le polynôme de Lagrange au point de Gauss z_ℓ sur l'intervalle $[0, 1]$ et restreint à l'intervalle $[0, 1]$. Ce schéma a la propriété d'être superconvergent en temps long (voir chapitre 3).

6.3 Schéma SLDG sur maillage non uniforme

Le schéma SLDG peut être adapté sur un maillage non uniforme. Les termes des matrices de masse $\int_{\mathbb{R}} \varphi_\ell(\alpha+s)\varphi_j(s)ds$ et $\int_{\mathbb{R}} \varphi_\ell(\alpha+s-1)\varphi_j(s)ds$ dans le cas uniforme ne sont plus des polynômes en α dans le cas d'un maillage non uniforme. Nous les évaluerons avec la formule de quadrature de Gauss.

Nous considérons toujours l'équation d'advection linéaire à vitesse constante (6.2.1) où le maillage est supposé maintenant non uniforme avec des pas d'espace valant :

$$\Delta x_{i+1/2} = x_{i+1} - x_i, \qquad i = 0, ..., N-1.$$

Le pas de temps Δt et les pas d'espace $\Delta x_{i+1/2}$ sont supposés constants en temps. Des conditions périodiques sont utilisées. Nous suivons la description du schéma SLDG donnée dans [25] en l'adaptant au cas non uniforme.

Notons $f^n(x) = f(t^n, x)$. En suivant les caractéristiques, nous avons pour toute fonction φ :

$$\int_{x_i}^{x_{i+1}} f^{n+1}(x)\varphi(x)dx = \int_{x_i}^{x_{i+1}} f^n(x - a\Delta t)\varphi(x)dx \tag{6.3.1}$$

$$= \int_{x_i - a\Delta t}^{x_{i+1} - a\Delta t} f^n(x)\varphi(x + a\Delta t)dx \tag{6.3.2}$$

Soit $d \in \mathbb{N}$. Sur chaque cellule $C_i = [x_i, x_{i+1}]$, nous plaçons $d+1$ points de Gauss notés par $\{x_{ij}\}_{(i,j)\in\{0,...,N-1\}\times\{0,...,d\}}$. De plus, nous notons par $\{z_j\}_{j\in\{0,...,d\}}$ les points de Gauss sur l'intervalle $[0,1]$ et $\{\omega_j\}_{j\in\{0,...,d\}}$ leurs poids correspondants. Nous considérons les polynômes de Lagrange aux points z_j restreint à l'intervalle $[0,1]$:

$$\varphi_j(x) = \prod_{\ell,\ell\neq j} \frac{x - z_\ell}{z_j - z_\ell} \text{ pour } x \in [0,1], \qquad \varphi_j(x) = 0 \text{ sinon}$$

et le polynôme correspondant sur la cellule C_i :

$$\varphi_{ij}(x) = \varphi_j\left(\frac{x - x_i}{\Delta x_{i+1/2}}\right).$$

Nous décomposons f^n sous la forme

$$f^n(x) = \sum_{i,j} f_{ij}^n \varphi_{ij}(x).$$

En injectant cette décomposition dans (6.3.2), le membre de gauche devient :

$$\begin{aligned}
\int_{x_i}^{x_{i+1}} f^{n+1}(x)\varphi_{i,j}(x)dx &= \int_{x_i}^{x_{i+1}} \sum_{i'=0}^{N-1}\sum_{j'=0}^{d} f_{i',j'}^{n+1}\varphi_{i',j'}(x)\varphi_{i,j}(x)dx \\
&= \int_{x_i}^{x_{i+1}} f_{i,j}^{n+1}\varphi_{i,j}^2(x)dx \\
&= f_{i,j}^{n+1}\Delta x_i \int_0^1 \varphi_{i,j}^2(y\Delta x_{i+1/2} + x_i)dy \\
&= f_{i,j}^{n+1}\Delta x_i \sum_{j'=0}^{d} \omega_{j'}\varphi_{i,j}^2(z_{j'}\Delta x_{i+1/2} + x_i) \\
&= f_{i,j}^{n+1}\Delta x_i\ \omega_j.
\end{aligned}$$

Nous notons par $i^\star$ l'entier tel que $x_i - a\Delta t \in [x_{i^\star}, x_{i^\star+1}[$ et par $\alpha_i \in [0,1[$ le réel défini par :

$$\alpha_i \Delta x_{i^\star+1/2} = (x_i - a\Delta t) - x_{i^\star}.$$

Ces notations sont illustrées dans la figure 6.1.

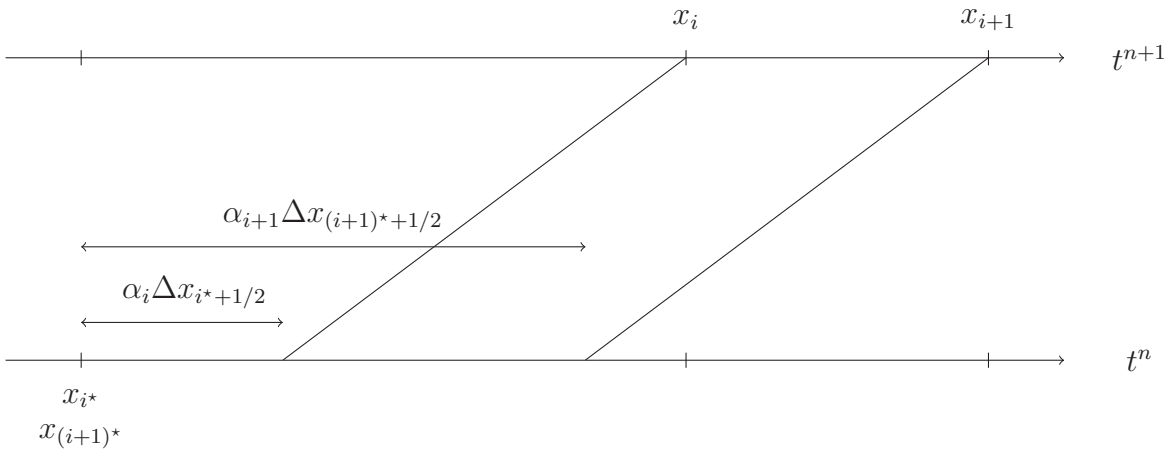


FIGURE 6.1 – Notations $i^\star$ et α_i

Grâce à ces notations, nous pouvons évaluer le membre de droite de la relation (6.3.2) :

$$\begin{aligned}
\int_{x_i - a\Delta t}^{x_{i+1}-a\Delta t} f^n(x)\varphi_{i,j}(x+a\Delta t)dx &= \int_{x_i - a\Delta t}^{x_{i+1}-a\Delta t} \sum_{i'=0}^{N-1}\sum_{j'=0}^{d} f^n_{i',j'}\varphi_{i',j'}(x)\varphi_{i,j}(x+a\Delta t)dx \\
&= \int_{x_{i^\star}+\alpha_i\Delta x_{i^\star+1/2}}^{x_{(i+1)^\star}+\alpha_{i+1}\Delta x_{(i+1)^\star+1/2}} \sum_{i'=0}^{N-1}\sum_{j'=0}^{d} f^n_{i',j'}\varphi_{i',j'}(x)\varphi_{i,j}(x+a\Delta t)dx \\
&= \int_{x_{i^\star}+\alpha_i\Delta x_{i^\star+1/2}}^{x_{i^\star+1}} \sum_{j'=0}^{d} f^n_{i^\star,j'}\varphi_{i^\star,j'}(x)\varphi_{i,j}(x+a\Delta t)dx \\
&\quad + \sum_{k=i^\star+1}^{(i+1)^\star-1} \int_{x_k}^{x_{k+1}} \sum_{j'=0}^{d} f^n_{k,j'}\varphi_{k,j'}(x)\varphi_{i,j}(x+a\Delta t)dx \\
&\quad + \int_{x_{(i+1)^\star}}^{x_{(i+1)^\star}+\alpha_{i+1}\Delta x_{(i+1)^\star+1/2}} \sum_{j'=0}^{d} f^n_{(i+1)^\star,j'}\varphi_{(i+1)^\star,j'}(x)\varphi_{i,j}(x+a\Delta t)dx
\end{aligned}$$

Comme dans le cas uniforme, nous cherchons une formulation pour les 3 termes de la dernière

expression :

$$\int_{x_{i^\star}+\alpha_i\Delta x_{i^\star+1/2}}^{x_{i^\star+1}} \sum_{j'=0}^{d} f^n_{i^\star,j'}\varphi_{i^\star,j'}(x)\varphi_{i,j}(x+a\Delta t)dx \tag{6.3.3}$$

$$\sum_{k=i^\star+1}^{(i+1)^\star-1} \int_{x_k}^{x_{k+1}} \sum_{j'=0}^{d} f^n_{k,j'}\varphi_{k,j'}(x)\varphi_{i,j}(x+a\Delta t)dx \tag{6.3.4}$$

$$\int_{x_{(i+1)^\star}}^{x_{(i+1)^\star}+\alpha_{i+1}\Delta x_{(i+1)^\star+1/2}} \sum_{j'=0}^{d} f^n_{(i+1)^\star,j'}\varphi_{(i+1)^\star,j'}(x)\varphi_{i,j}(x+a\Delta t)dx \tag{6.3.5}$$

sous la forme d'une matrice de masse faisant intervenir les intégrales de la forme

$$\int_0^1 \varphi_{j'}(Au)\varphi_j\left(Bu+C\right)du$$

où A, B et C sont à déterminer.

Etude du terme (6.3.3)

En faisant le changement de variable $s = \frac{x-x_{i^\star}}{\Delta x_{i^\star+1/2}}$, nous obtenons

$$\begin{aligned}
&\int_{x_{i^\star}+\alpha_i\Delta x_{i^\star+1/2}}^{x_{i^\star+1}} \sum_{j'=0}^{d} f^n_{i^\star,j'}\varphi_{i^\star,j'}(x)\varphi_{i,j}(x+a\Delta t)dx \\
&= \Delta x_{i^\star+1/2}\sum_{j'=0}^{d} f^n_{i^\star,j'}\int_{\alpha_i}^1 \varphi_{j'}(s)\varphi_j\left((s-\alpha_i)\frac{\Delta x_{i^\star+1/2}}{\Delta x_{i+1/2}}\right)ds.
\end{aligned}$$

Alors, le changement de variable $u = \frac{s-\alpha_i}{1-\alpha_i}$, conduit à

$$\begin{aligned}
&\Delta x_{i^\star}\sum_{j'=0}^{d} f^n_{i^\star,j'}\int_{\alpha_i}^1 \varphi_{j'}(s)\varphi_j\left((s-\alpha_i)\frac{\Delta x_{i^\star+1/2}}{\Delta x_{i+1/2}}\right)ds \\
&= (1-\alpha_i)\Delta x_{i^\star+1/2}\sum_{j'=0}^{d} f^n_{i^\star,j'}\int_0^1 \varphi_{j'}(u(1-\alpha_i)+\alpha_i)\varphi_j\left(u(1-\alpha_i)\frac{\Delta x_{i^\star+1/2}}{\Delta x_{i+1/2}}\right)du \\
&= (1-\alpha_i)\Delta x_{i^\star+1/2}\sum_{j'=0}^{d} f^n_{i^\star,j'}\int_0^1 \varphi_{j'}(A^{(1)}u+D^{(1)})\varphi_j\left(B^{(1)}u\right)du.
\end{aligned}$$

avec $A^{(1)} = (1-\alpha_i)$, $B^{(1)} = (1-\alpha_i)\frac{\Delta x_{i^\star+1/2}}{\Delta x_{i+1/2}}$ et $D^{(1)} = \alpha_i$.

Etude du terme (6.3.4)

En faisant le changement de variable $s = \frac{x-x_k}{\Delta x_{k+1/2}}$, nous obtenons

$$\sum_{k=i^\star+1}^{(i+1)^\star-1} \int_{x_k}^{x_{k+1}} \sum_{j'=0}^{d} f^n_{k,j'} \varphi_{k,j'}(x) \varphi_{i,j}(x + a\Delta t) dx$$

$$= \sum_{k=i^\star+1}^{(i+1)^\star-1} \Delta x_{k+1/2} \sum_{j'=0}^{d} f^n_{k,j'} \int_0^1 \varphi_{j'}(s) \varphi_j \left(s \frac{\Delta x_{k+1/2}}{\Delta x_{i+1/2}} + \frac{x_k - (x_{i^\star} + \alpha_i \Delta x_{i^\star+1/2})}{\Delta x_{i+1/2}} \right) ds$$

$$= \sum_{k=i^\star+1}^{(i+1)^\star-1} \Delta x_{k+1/2} \sum_{j'=0}^{d} f^n_{k,j'} \int_0^1 \varphi_{j'}(A^{(2)} u) \varphi_j \left(B^{(2)} u + C^{(2)} \right) du.$$

avec $A^{(2)} = 1$, $B^{(2)} = \frac{\Delta x_{k+1/2}}{\Delta x_{i+1/2}}$ et $C^{(2)} = \frac{x_k - (x_{i^\star} + \alpha_i \Delta x_{i^\star+1/2})}{\Delta x_{i+1/2}}$.

Etude du terme (6.3.5)

En faisant le changement de variable $s = \frac{x - x_{(i+1)^\star}}{\Delta x_{(i+1)^\star+1/2}}$, nous obtenons

$$\int_{x_{(i+1)^\star}}^{x_{(i+1)^\star} + \alpha_{i+1} \Delta x_{(i+1)^\star+1/2}} \sum_{j'=0}^{d} f^n_{(i+1)^\star,j'} \varphi_{(i+1)^\star,j'}(x) \varphi_{i,j}(x + a\Delta t) dx$$

$$= \Delta x_{(i+1)^\star+1/2} \sum_{j'=0}^{d} f^n_{(i+1)^\star,j'} \int_0^{\alpha_{i+1}} \varphi_{j'}(s) \varphi_j \left(s \frac{\Delta x_{(i+1)^\star+1/2}}{\Delta x_{i+1/2}} + \frac{x_{(i+1)^\star} - (x_{i^\star} + \alpha_i \Delta x_{i^\star+1/2})}{\Delta x_{i+1/2}} \right) ds.$$

Alors, le changement de variable $u = \frac{s}{\alpha_{i+1}}$, conduit à

$$\Delta x_{(i+1)^\star+1/2} \sum_{j'=0}^{d} f^n_{(i+1)^\star,j'} \int_0^{\alpha_{i+1}} \varphi_{j'}(s) \varphi_j \left(s \frac{\Delta x_{(i+1)^\star}}{\Delta x_i} + \frac{x_{(i+1)^\star-1/2} - (x_{i^\star-1/2} + \alpha_i \Delta x_{i^\star})}{\Delta x_i} \right) ds$$

$$= \alpha_{i+1} \Delta x_{(i+1)^\star+1/2} \sum_{j'=0}^{d} f^n_{(i+1)^\star,j'} \int_0^1 \varphi_{j'}(u\alpha_{i+1}) \times$$

$$\varphi_j \left(u\alpha_{i+1} \frac{\Delta x_{(i+1)^\star+1/2}}{\Delta x_{i+1/2}} + \frac{x_{(i+1)^\star} - (x_{i^\star} + \alpha_i \Delta x_{i^\star+1/2})}{\Delta x_{i+1/2}} \right) du$$

$$= \alpha_{i+1} \Delta x_{(i+1)^\star+1/2} \sum_{j'=0}^{d} f^n_{(i+1)^\star,j'} \int_0^1 \varphi_{j'}(A^{(3)} u) \varphi_j \left(B^{(3)} u + C^{(3)} \right) du.$$

avec $A^{(3)} = \alpha_{i+1}$, $B^{(3)} = \alpha_{i+1} \frac{\Delta x_{(i+1)^\star+1/2}}{\Delta x_{i+1/2}}$ et $C^{(3)} = \frac{x_{(i+1)^\star} - (x_{i^\star} + \alpha_i \Delta x_{i^\star+1/2})}{\Delta x_{i+1/2}}$.

Finalement, nous obtenons une formulation explicite pour le schéma SLDG dans le

cas non uniforme :

$$
\begin{aligned}
f_{i,j}^{n+1} &= \frac{1}{\omega_j}\Bigg((1-\alpha_i)\frac{\Delta x_{i^\star+1/2}}{\Delta x_{i+1/2}}\sum_{j'=0}^{d} f_{i^\star,j'}^{n}\int_0^1 \varphi_{j'}(A^{(1)}u+D^{(1)})\varphi_j\left(B^{(1)}u\right)du \\
&+ \sum_{k=i^\star}^{(i+1)^\star-1}\frac{\Delta x_{k+1/2}}{\Delta x_{i+1/2}}\sum_{j'=0}^{d} f_{k,j'}^{n}\int_0^1 \varphi_{j'}(A^{(2)}u)\varphi_j\left(B^{(2)}u+C^{(2)}\right)du \\
&+ \alpha_{i+1}\frac{\Delta x_{(i+1)^\star+1/2}}{\Delta x_{i+1/2}}\sum_{j'=0}^{d} f_{(i+1)^\star,j'}^{n}\int_0^1 \varphi_{j'}(A^{(3)}u)\varphi_j\left(B^{(3)}u+C^{(3)}\right)du\Bigg).
\end{aligned}
$$

Afin de calculer les intégrales intervenant dans les matrices de masse, nous utilisons la formule de quadrature de Gauss. Plus précisément, nous avons pour tout j, j' :

$$
\int_0^1 \varphi_{j'}(Au)\varphi_j\left(Bu+C\right)du = \sum_{k=0}^{d}\omega_k\varphi_{j'}(Az_k)\varphi_j\left(Bz_k+C\right).
$$

6.4 Résultats numériques

Dans le cas des ondes KEEN, une haute résolution de l'espace des phases est nécessaire pour les vitesses autour de ω/k. Nous avons choisi un maillage non uniforme simple constitué d'un maillage uniforme grossier et d'une zone uniforme raffinée tel que décrit sur la Figure 6.2. Les tailles des cellules sur les grilles grossière et fine valent

$$
\Delta v_{\text{coarse}} = \frac{v_{\max}-v_{\min}}{N_{\text{coarse}}},\ \Delta v_{\text{fine}} = \frac{v_{\max}-v_{\min}}{N_{\text{fine}}}
$$

où N_{fine} est un multiple de N_{coarse}. La zone raffinée est choisie avec $0 \le i_1 < i_2 \le N_{\text{coarse}}$ et le nombre total de cellules vaut

$$
N = i_1 + N_f + N_{\text{coarse}} - i_2,\ N_f = \frac{N_{\text{fine}}}{N_{\text{coarse}}}(i_2-i_1).
$$

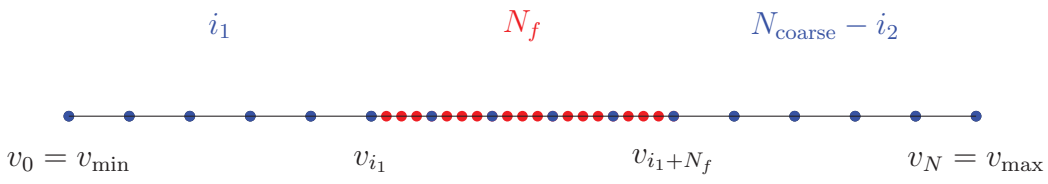


FIGURE 6.2 – Maillage simple non uniforme

Les figures 6.3 présentent la variation de la fonction de distribution $f(1000, x, v) - f_0(x, v)$ au temps $T = 1000$. Le maillage utilisé en espace est uniforme et le schéma d'advection utilisé est le schéma de Lagrange 17 pour les deux figures. Nous avons utilisé un maillage raffiné en vitesse puisque les ondes KEEN sont localisées autour de la vitesse du drive. Pour la

figure du haut, le schéma SLDG non uniforme est utilisé en vitesse alors que les splines cubiques non uniformes sont utilisées pour la figure du bas. Les résultats sont très proches et l'on n'observe pas de décalage malgré le temps élevé ($T = 1000$). Les figures 6.4 présentent les valeurs absolues des 3 premiers modes de Fourier de ρ en fonction du temps. Comme précédemment, le maillage utilisé en espace est uniforme et le schéma d'advection utilisé est Lagrange 17 pour les deux figures ; en vitesse, pour la figure du haut, le schéma SLDG non uniforme est utilisé alors que les splines cubiques non uniformes sont utilisées pour la figure du bas. On observe une bonne adéquation des modes de Fourrier par rapport à la solution convergée (LAG17 uniforme en espace et en vitesse, $N_x = N_v = 2048$, sur GPU). A partir du temps $T = 400$, des différences commencent à apparaître et sont plus marquées à partir du temps $T = 500$. De nouveau, les deux schémas non uniformes donnent des résultats très proches.

6.5 Conclusion

Le schéma SLDG non uniforme, qui est d'ordre élevé, donne des résultats similaires par rapport à l'implémentation par splines cubiques non uniformes préexistante, ce qui valide cette méthode. Ce schéma admet une propriété de superconvergence en temps long pour l'équation d'advection linéaire à vitesse constante. Le solveur de Vlasov-Poisson $1D \times 1D$ non uniforme permet de réduire le nombre de points ; ceci est encourageant pour les futures simulations $2D \times 2D$. Un travail permettant l'accélération du code non uniforme SLDG est envisagé avec à terme une intégration dans la librairie SELALIB.

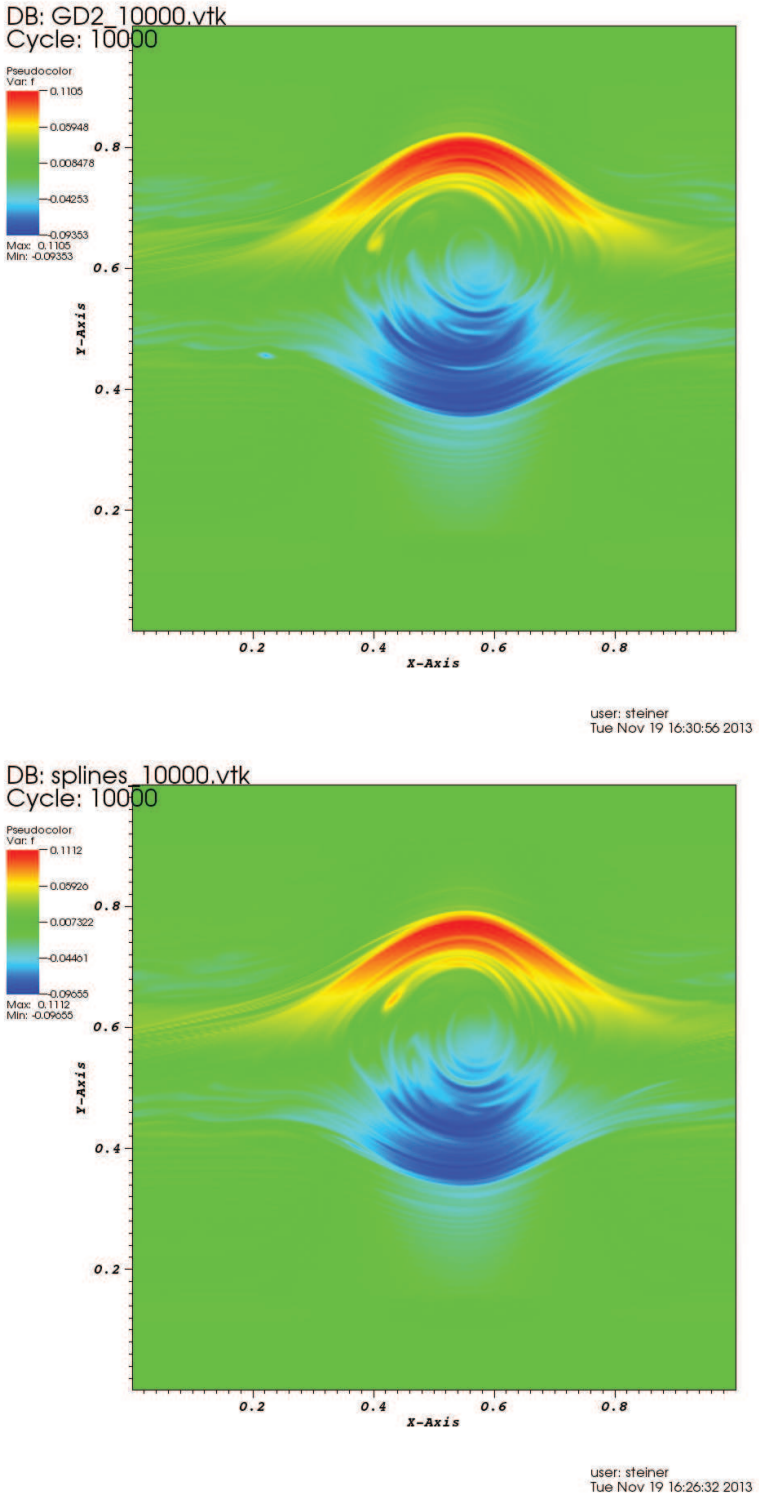


FIGURE 6.3 – $f(1000, x, v) - f_0(x, v)$, $\Delta t = 0.1$. Solution sur un maillage uniforme en espace (LAG17, $N_x = 256$) et un maillage uniforme raffiné en vitesse avec (en haut) le schéma SLDG non uniforme, $d = 2$ et $N_v = 374$ ($N_{\text{coarse}} = 64$, $N_{\text{fine}} = 2048$, $i_1 = 34$, $i_2 = 44$) (en bas) splines cubiques non uniformes et $N_v = 374 \times 3$ ($N_{\text{coarse}} = 64 \times 3$, $N_{\text{fine}} = 2048 \times 3$, $i_1 = 34 \times 3$, $i_2 = 44 \times 3$)

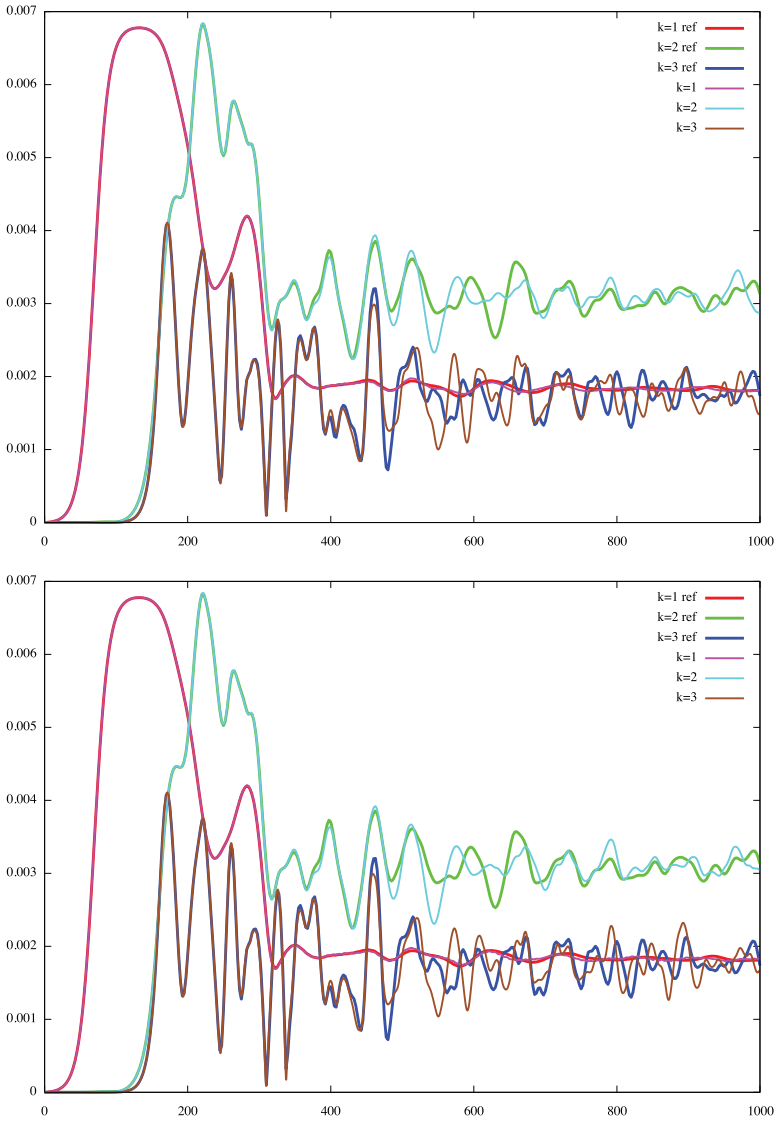


FIGURE 6.4 – Valeurs absolues des 3 premiers modes de Fourier de ρ par rapport au temps. Solution de référence avec LAG17 $N_x = N_v = 2048$ sur GPU en double précision (rouge, vert et bleu) comparé à la solution sur un maillage uniforme en espace (LAG17, $N_x = 256$) et un maillage uniforme raffiné en vitesse. avec (en haut) schéma SLDG non uniforme, $d = 2$ et $N_v = 374$ ($N_{\text{coarse}} = 64$, $N_{\text{fine}} = 2048$, $i_1 = 34$, $i_2 = 44$) (en bas) splines cubiques non uniformes et $N_v = 374 \times 3$ ($N_{\text{coarse}} = 64 \times 3$, $N_{\text{fine}} = 2048 \times 3$, $i_1 = 34 \times 3$, $i_2 = 44 \times 3$)

Troisième partie

Modèle gyrocinétique

L'effort de calcul pour résoudre numériquement le système de Vlasov-Maxwell en 6 dimensions décrivant la turbulence dans les plasmas de tokamak reste encore hors de portée pour les supercalculateurs actuels. Toutes les simulations numériques effectuées jusqu'à présent dans ce domaine prennent en considération le modèle gyrocinétique pour réduire ce problème d'une dimension. Cette considération prend en compte le fait que (i) les fluctuations électromagnétiques se produisent sur des échelles de temps beaucoup plus longues que la période de giration des particules chargées ($\omega/\Omega_c \ll 1$ avec ω la fréquence de fluctuation et Ω_c la fréquence cyclotronique), et (ii) la longueur d'onde de ces fluctuations est beaucoup plus petite que la grandeur caractéristique des gradients du champ magnétique, de la densité et de la température. Voir [89] pour un examen détaillé dans le cadre gyrocinétique et les simulations du transport turbulent dans les plasmas de fusion. Le modèle gyrocinétique peut être dérivé (voir [96]) en moyennant sur la giration rapide des particules chargées autour des lignes de champ magnétique. La configuration magnétique toroïdale considérée ici est simplifiée. En effet, les surfaces de flux magnétique sont supposées être des tores concentriques avec des sections transversales circulaires. Le nouveau jeu de coordonnées 5D correspond à : (i) les coordonnées de l'espace toroïdal 3D (r, θ, φ) (avec r la direction radiale, θ et φ l'angle poloïdal (resp. toroïdal)), et (ii) 2D dans l'espace des vitesses avec $v_\parallel$ la vitesse parallèle aux lignes de champ magnétique et $\mu = mv_\perp^2/(2B)$ le moment magnétique où $v_\perp$ représente la vitesse dans le plan orthogonal au champ magnétique. Il est important de noter que dans ce contexte μ est un invariant adiabatique, de sorte qu'il joue le rôle d'un paramètre dans l'équation de Vlasov 5D gyrocinétique.

Dans la suite, le problème 4D correspond au cas où l'on considère une valeur unique de μ, i.e. le même rayon de Larmor est considéré pour toutes les particules ($\mu = 0$ dans le chapitre 7, μ fixé quelconque dans la section 8.5.1 du chapitre 8). Dans le problème 5D que nous considérons dans la section 8.5.2 du chapitre 8 et dans le chapitre 9, plusieurs valeurs de μ sont considérées pour tenir compte de la dépendance du rayon de Larmor par rapport à $v_\perp$.

L'évolution en temps de la fonction de distribution f du centre guide est donnée par l'équation gyrocinétique conservative (voir aussi Eqs (17)-(20) dans [89]) :

$$B_\parallel^* \frac{\partial f}{\partial t} + \boldsymbol{\nabla} \cdot \left(B_\parallel^* \frac{\mathrm{d}\mathbf{x}_G}{\mathrm{d}t} f \right) + \frac{\partial}{\partial v_{G\parallel}} \left(B_\parallel^* \frac{\mathrm{d}v_{G\parallel}}{\mathrm{d}t} f \right) = 0 \tag{6.5.1}$$

où $\mathbf{x}_G$ et $v_{G\parallel}$ sont respectivement les coordonnées d'espace et la vitesse parallèle des centres guides. Dans la limite électrostatique, pour une particule de masse m et de charge q les équations du mouvement des centres guides sont données par

$$\frac{\mathrm{d}\mathbf{x}_G}{\mathrm{d}t} = v_{G\parallel}\mathbf{b}^* + \mathbf{v}_{E\times B} + \mathbf{v}_D \tag{6.5.2}$$

$$m\frac{\mathrm{d}v_{G\parallel}}{\mathrm{d}t} = -\mu \boldsymbol{\nabla}_\parallel^* B - q\boldsymbol{\nabla}_\parallel^* \bar{\Phi} + m v_{G\parallel} \mathbf{v}_{E\times B} \cdot \frac{\boldsymbol{\nabla} B}{B} \tag{6.5.3}$$

où $\boldsymbol{\nabla}_\parallel^* \equiv \mathbf{b}^* \cdot \boldsymbol{\nabla}$, tandis que $\mathbf{b}^*$ et $B_\parallel^*$ sont définis par :

$$\mathbf{b}^* \equiv \frac{\mathbf{B}}{B_\parallel^*} + \frac{m v_{G\parallel}}{q B_\parallel^* B} \boldsymbol{\nabla} \times \mathbf{B} \tag{6.5.4}$$

$$B_\parallel^* \equiv B + \frac{m v_{G\parallel}}{qB} \mathbf{b} \cdot (\boldsymbol{\nabla} \times \mathbf{B}) \,. \tag{6.5.5}$$

Le drift '$\mathbf{E}\times\mathbf{B}$' est égal à $\mathbf{v}_{E\times B} = (1/B_{\parallel}^{*})\mathbf{b}\times\boldsymbol{\nabla}\bar{\Phi}$ tandis que la courbure du drift est définie par $\mathbf{v}_D = \left(\frac{mv_{G\parallel}^2+\mu B}{qB_{\parallel}^{*}}\right)\mathbf{b}\times\frac{\boldsymbol{\nabla}B}{B}$.

La fonction en cinq dimensions ainsi obtenue doit être auto-cohérente couplée avec les équations de Maxwell. Dans la suite, nous considérons l'approximation électrostatique, où les équations de Maxwell sont réduites à une équation de quasi-neutralité qui est l'équivalent asymptotique de l'équation de Poisson. Etant donné que l'équation de Poisson est définie sur les coordonnées des particules, la résolution du système de Vlasov-Poisson gyrocinétique nécessite un opérateur qui transforme l'espace de phase gyrocentré dans l'espace des phases des particules. Cet opérateur est l'opérateur de gyromoyenne (noté $\mathcal{J}_{\sqrt{2\mu}}$) que nous détaillerons dans le chapitre 8. Nous référons à une abondante littérature concernant ce sujet (voir [85, 102, 105] ainsi les références contenues dans ces articles). Le potentiel électrostatique gyromoyenné $\mathcal{J}_{\sqrt{2\mu}}\Phi$ est la solution de l'équation de quasi-neutralité 3D auto-consistante couplée :

$$\frac{1}{T_i}(\Phi-\tilde{\Phi})+\frac{1}{T_e(r)}(\Phi-\langle\Phi\rangle)=\frac{1}{n_0(r)}\mathcal{J}_{\sqrt{2\mu}}\left(\int f-f_{eq}dv\right), \tag{6.5.6}$$
$$\tilde{\Phi}(\mathbf{x}):=\frac{1}{T_i(r)}\int_{\mathbb{R}^+}\mathcal{J}^2_{\sqrt{2\mu}}(\Phi)(\mathbf{x})e^{-\mu/T_i}d\mu,\qquad \langle\Phi\rangle(r,\theta)=\frac{1}{L}\int_0^L\Phi(r,\theta,z)dz$$

où T_i, T_e, n_0 sont des profils de température et de densité qui seront définis plus tard. Nous utiliserons également une seconde version de l'équation de quasi-neutralité

$$-\left(\partial_r^2\Phi+\left(\frac{1}{r}+\frac{\partial_r n_0(r)}{n_0(r)}\right)\partial_r\Phi+\frac{1}{r^2}\partial_\theta^2\Phi\right)+$$
$$\frac{1}{T_e(r)}(\Phi-\langle\Phi\rangle)=\frac{1}{n_0(r)}\mathcal{J}_{\sqrt{2\mu}}\left(\int f-f_{eq}dv\right). \tag{6.5.7}$$

Les liens entre ces deux équations seront effectués dans le chapitre 9.

Le modèle SLAB 4D est un modèle simplifié du système d'équations (6.5.1)-(6.5.7). Un plasma périodique cylindrique de rayon a et de longueur $2\pi R$ (avec R le grand rayon) est considéré comme cas limite d'un tore étiré. Le plasma est confiné par un fort champ magnétique uniforme $\mathbf{B}=B\mathbf{e}_z$ où $\mathbf{e}_z$ représente le vecteur unitaire dans la direction toroïdale z. Avec ces hypothèses, les drifts de vitesse sont réduits au drift $\mathbf{E}\times\mathbf{B}$. Ce cas SLAB 4D est équivalent à celui traité dans [81] ou [80]. L'équation satisfaite par la fonction de distribution des ions $f(t,r,\theta,z,v)$ suivant le mouvement du centre guide vaut :

$$\partial_t f-\left(\frac{\partial_\theta\mathcal{J}_{\sqrt{2\mu}}\Phi}{r}\right)\partial_r f+\left(\frac{\partial_r\mathcal{J}_{\sqrt{2\mu}}\Phi}{r}\right)\partial_\theta f+$$
$$v\partial_z f-\left(\partial_z\mathcal{J}_{\sqrt{2\mu}}\Phi\right)\partial_v f=0. \tag{6.5.8}$$

pour $(r,\theta,z,v)\in[r_{\min},r_{\max}]\times[0,2\pi]\times[0,L]\times[-v_{\max},v_{\max}]$. Le modèle Drift-Kinetic SLAB 4D est une simplification de ce modèle en imposant la condition $\mu=0$. Il décrit alors la dynamique des plasmas sans effet du rayon de Larmor sur les électrons (voir [80] et [81]).

Après avoir décrit les 3 opérateurs (équation de Vlasov, équation de quasi-neutralité et opérateur de gyromoyenne), l'objectif de cette partie est de proposer et de comparer des méthodes numériques pour ces opérateurs pour au moins le modèle SLAB 4D.

Chapitre 7

Interpolation de type Hermite

Dans ce chapitre, nous présentons un opérateur d'interpolation de type Hermite en géométrie polaire. La méthode semi-Lagrangienne BSL [6] permettant de résoudre l'équation d'advection 2D consiste en une interpolation de la fonction advectée au pied des caractéristiques. Nous comparerons l'interpolation d'Hermite à l'interpolation plus classique par splines cubiques pour la méthode BSL dans le cadre de la résolution numérique du modèle Drift-Kinetic 4D en géométrie SLAB.

7.1 Opérateur d'interpolation d'Hermite

7.1.1 Introduction

Nous présentons, dans cette section, l'opérateur d'interpolation d'Hermite. Cette méthode d'interpolation consiste à reconstruire une fonction polynomiale de degré 3 sur une cellule de telle sorte à ce que cette fonction polynomiale (resp. sa dérivée) coïncide avec les valeurs de la fonction à interpoler (resp. sa dérivée) aux bords de la cellule. Pour cela, les valeurs des dérivées sont reconstruites à partir des valeurs nodales de la fonction à interpoler par différences finies d'ordre d quelconque. La méthode d'interpolation d'Hermite reste d'ordre 3 ce qui en permet une utilisation souple. Malgré cette complexité peu importante, il est possible d'améliorer les performances de l'interpolation par l'augmentation de l'ordre arbitraire de reconstruction des dérivées. De plus, l'interpolation d'Hermite est locale ; elle ne nécessite que quelques points (selon le degré de reconstruction des dérivées) autour de la position cible de l'interpolation contrairement à l'approche par splines cubiques par exemple, mais en nécessitant tout de même plus de points que les splines locales [82]. Une reconstruction par interpolation d'Hermite a déjà été étudiée dans la section 2.2.2 du chapitre 2 (schémas LAG 3, PPM 0, PPM 1 et PPM 2) afin de reconstruire la primitive de la solution de l'équation d'advection sur chaque cellule.

7.1.2 Cas d'un maillage unidimentionnel

Nous détaillons ici l'opérateur d'interpolation d'Hermite dans le cas unidimentionnel avant d'aborder le cas polaire dans la prochaine partie. Ainsi, considérons le domaine $\Omega = [a, b] \subset \mathbb{R}$ divisé en N cellules :

$$C_i = [x_i, x_{i+1}], \quad i = 0, ..., N-1.$$

Nous supposons ici que le maillage est uniforme : le pas d'espace Δx satisfait

$$\Delta x = x_{i+1} - x_i = \frac{b-a}{N}, \qquad i = 0, ..., N-1.$$

Soit $\alpha \in [0, 1[$. La reconstruction de f par interpolation d'Hermite sur la cellule $\mathcal{C}_i$ s'écrit :

$$\begin{aligned} f(x_i + \alpha\Delta x) \;\approx\; & (2\alpha+1)(1-\alpha)^2 f(x_i) + \alpha^2(3-2\alpha) f(x_{i+1}) \\ & + \alpha(1-\alpha)^2 f'(x_i^+) + \alpha^2(\alpha-1) f'(x_{i+1}^-). \end{aligned}$$

Les dérivées à droite et à gauche aux interfaces des cellules sont reconstruites par différences finies d'ordre quelconque d :

$$f'(x_i^-)_{i=0..N} \approx \Psi_d^-(f(x_i)_{i=0..N}), \qquad f'(x_i^+)_{i=0..N} \approx \Psi_d^+(f(x_i)_{i=0..N})$$

où les opérateurs $\Psi_d^\pm : \mathbb{R}^{N+1} \to \mathbb{R}^{N+1}$ valent :

$$(\Psi_d^-(X))_i = \sum_{k=r_d^-}^{s_d^-} \omega_{k,d}^- X_{i+k}, \qquad (\Psi_d^+(X))_i = \sum_{k=r_d^+}^{s_d^+} \omega_{k,d}^+ X_{i+k}$$

avec

$$\omega_{k,d}^\pm = \prod_{j=r_d^\pm,\ j\neq k,0}^{s_d^\pm} (-j) \Big/ \prod_{j=r_d^\pm,\ j\neq k}^{s_d^\pm} (k-j)$$

pour $k \neq 0$ et

$$\omega_{0,d}^\pm = -\sum_{k=r_d^\pm,\ k\neq 0}^{s_d^\pm} \omega_{k,d}.$$

Pour une reconstruction d'ordre pair $d = 2p$, le stencil vaut :

$$r_d^- = -p, \qquad s_d^- = p, \qquad r_d^+ = -p+1, \qquad s_d^+ = p+1$$

et pour une reconstruction d'ordre impair $d = 2p+1$, nous avons :

$$r_d^- = r_d^+ = -p, \qquad s_d^- = s_d^+ = p+1.$$

Remarque 7.1.1.

(1) Pour $d = 3$, l'interpolation d'Hermite coïncide avec celle de Lagrange d'ordre 3.

(2) Nous observons que dans le cas de l'ordre impair, la reconstruction est décentrée et la fonction dérivée ainsi reconstruite sera de classe C^0.

7.1.3 Cas d'un maillage polaire

Considérons à présent un maillage polaire uniforme sur le domaine $[r_{\min}, r_{\max}] \times [0, 2\pi]$ comprenant $N_r \times N_\theta$ cellules :

$$C_{ij} = [r_i, r_{i+1}] \times [\theta_j, \theta_{j+1}], \qquad i = 0..N_r - 1,\ j = 0..N_\theta - 1$$

où

$$\begin{aligned} r_i &= r_{\min} + i\frac{r_{\max}-r_{\min}}{N_r}, \qquad i = 0, ..., N_r, \\ \theta_j &= j\frac{2\pi}{N_\theta}, \qquad j = 0, ..., N_\theta. \end{aligned}$$

L'interpolation d'Hermite sur maillage polaire est alors une succession d'interpolations d'Hermite unidimensionnelles. Plus précisément, pour évaluer $f(\tilde{r}, \tilde{\theta})$ où $(\tilde{r}, \tilde{\theta}) \in C_{ij}$, nous suivons l'algorithme suivant :

- Sur $[\theta_j, \theta_{j+1}]$, interpolation de la fonction $f(r_i, .)$ pour évaluer $f(r_i, \tilde{\theta})$.
- Sur $[\theta_j, \theta_{j+1}]$, interpolation de la fonction $f(r_{i+1}, .)$ pour évaluer $f(r_{i+1}, \tilde{\theta})$.
- Sur $[\theta_j, \theta_{j+1}]$, interpolation de la fonction $\partial_r f(r_i^+, .)$ pour évaluer $\partial_r f(r_i^+, \tilde{\theta})$.
- Sur $[\theta_j, \theta_{j+1}]$, interpolation de la fonction $\partial_r f(r_{i+1}^-, .)$ pour évaluer $\partial_r f(r_{i+1}^-, \tilde{\theta})$.
- Sur $[r_i, r_{i+1}]$, interpolation de la fonction $f(., \tilde{\theta})$ en utilisant les 4 évaluations précédentes pour calculer $f(\tilde{r}, \tilde{\theta})$.

Afin de réaliser ces interpolations 1D, nous construisons, dans un premier temps, les dérivées partielles aux interfaces des cellules :

$$\begin{aligned} (\partial_r f(r_i^+, \theta_j))_{i=0,...,N_r} &\approx \Psi_d^+(f(r_i, \theta_j)_{i=0,..,N_r}) \qquad \forall j = 0, .., N_\theta \\ (\partial_r f(r_i^-, \theta_j))_{i=0,...,N_r} &\approx \Psi_d^-(f(r_i, \theta_j)_{i=0,..,N_r}) \qquad \forall j = 0, .., N_\theta \\ (\partial_\theta f(r_i, \theta_j^+))_{j=0,...,N_\theta} &\approx \Psi_d^+(f(r_i, \theta_j)_{j=0,..,N_\theta}) \qquad \forall i = 0, .., N_r \\ (\partial_\theta f(r_i, \theta_j^-))_{j=0,...,N_\theta} &\approx \Psi_d^-(f(r_i, \theta_j)_{j=0,..,N_\theta}) \qquad \forall i = 0, .., N_r \end{aligned}$$

puis les dérivées secondes :

$$\begin{aligned} (\partial_{r,\theta} f(r_i^+, \theta_j^+))_{i=0,...,N_r} &\approx \Psi_d^+(\partial_\theta f(r_i, \theta_j^+)_{i=0,..,N_r}) \qquad \forall j = 0, .., N_\theta \\ (\partial_{r,\theta} f(r_i^-, \theta_j^+))_{i=0,...,N_r} &\approx \Psi_d^-(\partial_\theta f(r_i, \theta_j^+)_{i=0,..,N_r}) \qquad \forall j = 0, .., N_\theta \\ (\partial_{r,\theta} f(r_i^+, \theta_j^-))_{i=0,...,N_r} &\approx \Psi_d^+(\partial_\theta f(r_i, \theta_j^-)_{i=0,..,N_r}) \qquad \forall j = 0, .., N_\theta \\ (\partial_{r,\theta} f(r_i^-, \theta_j^-))_{i=0,...,N_r} &\approx \Psi_d^-(\partial_\theta f(r_i, \theta_j^-)_{i=0,..,N_r}) \qquad \forall j = 0, .., N_\theta. \end{aligned}$$

Nous noterons PPM(p) l'opérateur d'interpolation d'Hermite polaire pour une reconstruction des dérivées de degré pair : $d = 2p$ et LAGH(d) dans le cas d'une reconstruction de degré impair : $d = 2p + 1$. Cet opérateur sera utilisé dans la section suivante ainsi que pour le calcul de l'opérateur de gyromoyenne (chapitre 8).

7.2 Modèle Drift-Kinetic SLAB 4D et méthode de splitting

Le modèle Drift-Kinetic SLAB 4D décrit la dynamique des plasmas sans effet du rayon de Larmor. L'équation satisfaite par la fonction de distribution $f(t, r, \theta, z, v)$ suivant le mouvement du centre guide vaut :

$$\partial_t f - \frac{\partial_\theta \Phi}{r}\partial_r f + \frac{\partial_r \Phi}{r}\partial_\theta f + v\partial_z f - \partial_z \Phi \partial_v f = 0 \tag{7.2.1}$$

pour $(r,\theta,z,v) \in [r_{\min}, r_{\max}] \times [0,2\pi] \times [0,L] \times [-v_{\max}, v_{\max}]$. Le potentiel auto-consistant $\Phi = \Phi(r,\theta,z)$ résout l'équation de quasi-neutralité

$$-\left(\partial_r^2\Phi + \left(\frac{1}{r} + \frac{\partial_r n_0(r)}{n_0(r)}\right)\partial_r\Phi + \frac{1}{r^2}\partial_\theta^2\Phi\right) + \frac{1}{T_e(r)}(\Phi - \lambda\langle\Phi\rangle) = \frac{1}{n_0(r)}\int_{\mathbb{R}} f dv - 1, \quad (7.2.2)$$

$$\langle\Phi\rangle(r,\theta) = \frac{1}{L}\int_0^L \Phi(r,\theta,z)dz$$

où λ vaut 1 ou 0 suivant que l'on considère le terme $\langle\Phi\rangle$ ou non (avec/sans zonal flow). Afin de traiter le cas 4D, une méthode classique consiste à effectuer un splitting directionnel (voir [80], [86]) :

- Résolution de l'équation 1D d'advection $\partial_t f + v\partial_z f = 0$ sur $\Delta t/2$ par une méthode semi-Lagrangienne avec interpolation par splines cubiques.
- Résolution de l'équation 1D d'advection $\partial_t f - \partial_z\Phi\partial_v f = 0$ sur $\Delta t/2$ par une méthode semi-Lagrangienne avec interpolation par splines cubiques.
- Résolution de l'équation de quasi-neutralité par FFT en θ, z et différences finies d'ordre 2 en r.
- Calcul des dérivées $(\partial_r\Phi, \partial_\theta\Phi, \partial_z\Phi)$ par splines cubiques en r,θ et différences finies d'ordre 2 en z.
- Résolution de l'advection 2D en (r,θ).
- Résolution de l'équation 1D d'advection $\partial_t f - \partial_z\Phi\partial_v f = 0$ sur $\Delta t/2$ par une méthode semi-Lagrangienne avec interpolation par splines cubiques.
- Résolution de l'équation 1D d'advection $\partial_t f + v\partial_z f = 0$ sur $\Delta t/2$ par une méthode semi-Lagrangienne avec interpolation par splines cubiques.

Ce splitting est d'ordre 2 en temps (voir [80] pour une preuve formelle). Pour l'étape d'advection 2D, nous utilisons la méthode semi-Lagrangienne classique BSL [6] qui consiste à calculer le pied des caractéristiques et interpoler la fonction de distribution en ces points. Pour cette dernière étape d'interpolation, nous utilisons soit les splines cubiques, soit l'interpolateur d'Hermite (PPM et LAGH). Une méthode alternative pour l'étape d'advection 2D est proposée dans [77], permettant une meilleure conservation de la masse et de l'énergie totale. Voir également [78, 79] pour une autre reconstruction 2D appliquée au centre-guide et au modèle Drift-Kinetic.

7.3 Résultats numériques

Nous résolvons le système d'équations (7.2.1)-(7.2.2) par la méthode de splitting décrite ci-dessus en utilisant la plate-forme SELALIB [67]. La fonction de distribution initiale est donnée par

$$f(t=0,r,\theta,z,v) = f_{eq}(r,v)\left[1 + \varepsilon\exp\left(-\frac{(r-r_p)^2}{\delta r}\right)\cos\left(\frac{2\pi n}{L}z + m\theta\right)\right],$$

où la fonction d'équilibre vaut :

$$f_{eq}(r,v) = \frac{n_0(r)\exp\left(-\frac{v^2}{2T_i(r)}\right)}{(2\pi T_i(r))^{1/2}}.$$

Les profils $\{T_i, T_e, n_0\}$ ont pour expression analytique :

$$\mathcal{P}(r) = \mathcal{C}_{\mathcal{P}} \exp\left(-\kappa_{\mathcal{P}} \delta r_{\mathcal{P}} \tanh\left(\frac{r - r_p}{\delta r_{\mathcal{P}}}\right)\right), \quad \mathcal{P} \in \{T_i, T_e, n_0\},$$

avec les constantes

$$C_{T_i} = C_{T_e} = 1,\ C_{n_0} = \frac{r_{\max} - r_{\min}}{\int_{r_{\min}}^{r_{\max}} \exp\left(-\kappa_{n_0} \delta r_{n_0} \tanh\left(\frac{r - r_{\mathcal{P}}}{\delta r_{n_0}}\right)\right) dr}.$$

Les paramètres sont ceux du cas MEDIUM de [86]

$$\begin{aligned} r_{\min} = 0.1,\ r_{\max} = 14.5, \\ \kappa_{n_0} = 0.055,\ \kappa_{T_i} = \kappa_{T_e} = 0.27586, \\ \delta r_{T_i} = \delta r_{T_e} = \frac{\delta r_{n_0}}{2} = 1.45,\ \varepsilon = 10^{-6},\ n = 1,\ m = 5, \\ L = 1506.759067,\ r_p = \frac{r_{\min} + r_{\max}}{2},\ \delta r = \frac{4\delta r_{n_0}}{\delta r_{T_i}}. \end{aligned}$$

Nous nous intéressons à l'évolution en temps de la masse totale :

$$\mathcal{M}(t) = \int_{r_{\min}}^{r_{\max}} \int_0^{2\pi} \int_0^L \int_{\mathbb{R}} f(t, r, \theta, z, v) r dv dz d\theta dr,$$

de l'énergie totale :

$$\begin{aligned} \mathcal{E}(t) \ &= \int_{r_{\min}}^{r_{\max}} \int_0^{2\pi} \int_0^L \int_{\mathbb{R}} \frac{v^2}{2} f(t, r, \theta, z, v) r dv dz d\theta dr \\ &+ \int_{r_{\min}}^{r_{\max}} \int_0^{2\pi} \int_0^L \int_{\mathbb{R}} f(t, r, \theta, z, v) \Phi(t, r, \theta, z) r dv dz d\theta dr, \end{aligned}$$

et pour $p = 1, 2$, de la norme L^p :

$$\|f\|_{L^p}^p(t) = \int_{r_{\min}}^{r_{\max}} \int_0^{2\pi} \int_0^L \int_{\mathbb{R}} |f(t, r, \theta, z, v)|^p r dv dz d\theta dr.$$

Des coupes poloïdales de la fonction de distribution sont présentées (Fig. 7.1 et 7.2) pour différentes méthodes d'interpolation dans l'étape d'advection 2D. Il apparaît que les méthodes par splines cubiques ou PPM développent des oscillations numériques lorsque le pas de temps diminue (Fig. 7.2), ce qui n'est pas le cas pour les interpolations LAGH (Fig. 7.1) qui sont plus diffusives. Par ailleurs, lorsque l'on augmente le degré de reconstruction des dérivées dans le cas LAGH, on voit l'apparition de structures plus fines (Fig. 7.1). La Figure 7.3 présente l'évolution en temps de quantités conservées au niveau continu (normes L^1 et L^2, masse et énergie totale). Il apparaît que les méthodes PPM conservent le mieux ces quantités, suivi des splines cubiques et enfin des méthodes LAGH. Lorsque le pas de temps diminue, la norme L^2 est mieux conservée dans le cas des splines cubiques (Fig. 7.4), ce qui est en accord avec le fait que cette méthode diffuse peu et crée des oscillations. Ce phénomène est moins marqué pour l'interpolation LAG3. Concernant les temps de calcul

(Table 8.12), la méthode avec interpolation par splines cubiques est plus rapide que celle avec interpolation d'Hermite. Pour Hermite, l'augmentation du degré de la reconstruction des dérivées augmente légèrement le temps de calcul.

Ces résultats sont à relier avec les Figures 4.10 et 4.11 du chapitre 4. Sur ces figures, nous observons également que les oscillations de la reconstruction centrée sont accentuées lorsque l'on diminue le pas de temps, contrairement aux reconstructions décentrées. Cette similarité pourrait s'expliquer par la présence d'un mouvement d'advection circulaire à coefficient constant dans le cas du modèle Drift-Kinetic.

7.4 Conclusion

Dans ce chapitre, nous avons présenté un opérateur d'interpolation de type Hermite que nous avons utilisé pour la méthode BSL dans le cadre de la résolution numérique du modèle Drift-Kinetic 4D en géométrie SLAB. Il apparaît que l'influence de cet opérateur est radicalement différente suivant que la reconstruction des dérivées soit centrée ou décentrée. Dans le cas d'une reconstruction centrée (PPM), le comportement est assez proche de celui observé avec l'interpolation par splines cubiques, caractérisé par une bonne conservation des quantités physiques et par l'apparition d'oscillations numériques lorsque le pas de temps décroît. En considérant une reconstruction décentrée (LAGH) le schéma est plus diffusif et crée moins d'oscillations numériques pour de petits pas de temps.

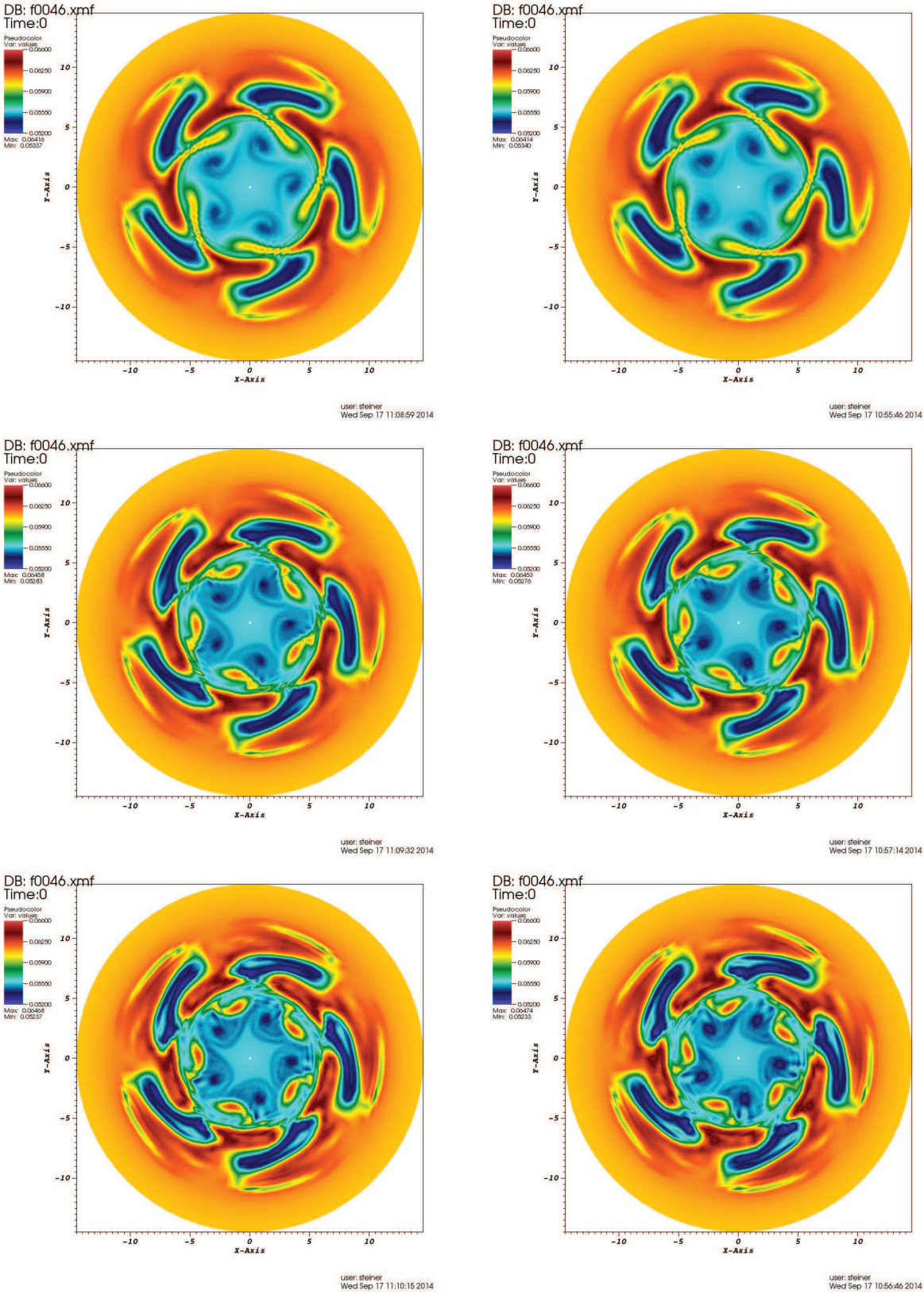


FIGURE 7.1 – Coupe poloïdale $f(r, \theta, z = 0, v_{\parallel} = 0)$ pour $64 \times 64 \times 32 \times 64$, $T = 4600$ avec $\Delta t = 4$ (à gauche) et $\Delta t = 2$ (à droite) ; LAG3 (en haut), LAGH5 (au milieu) et LAGH9 (en bas).

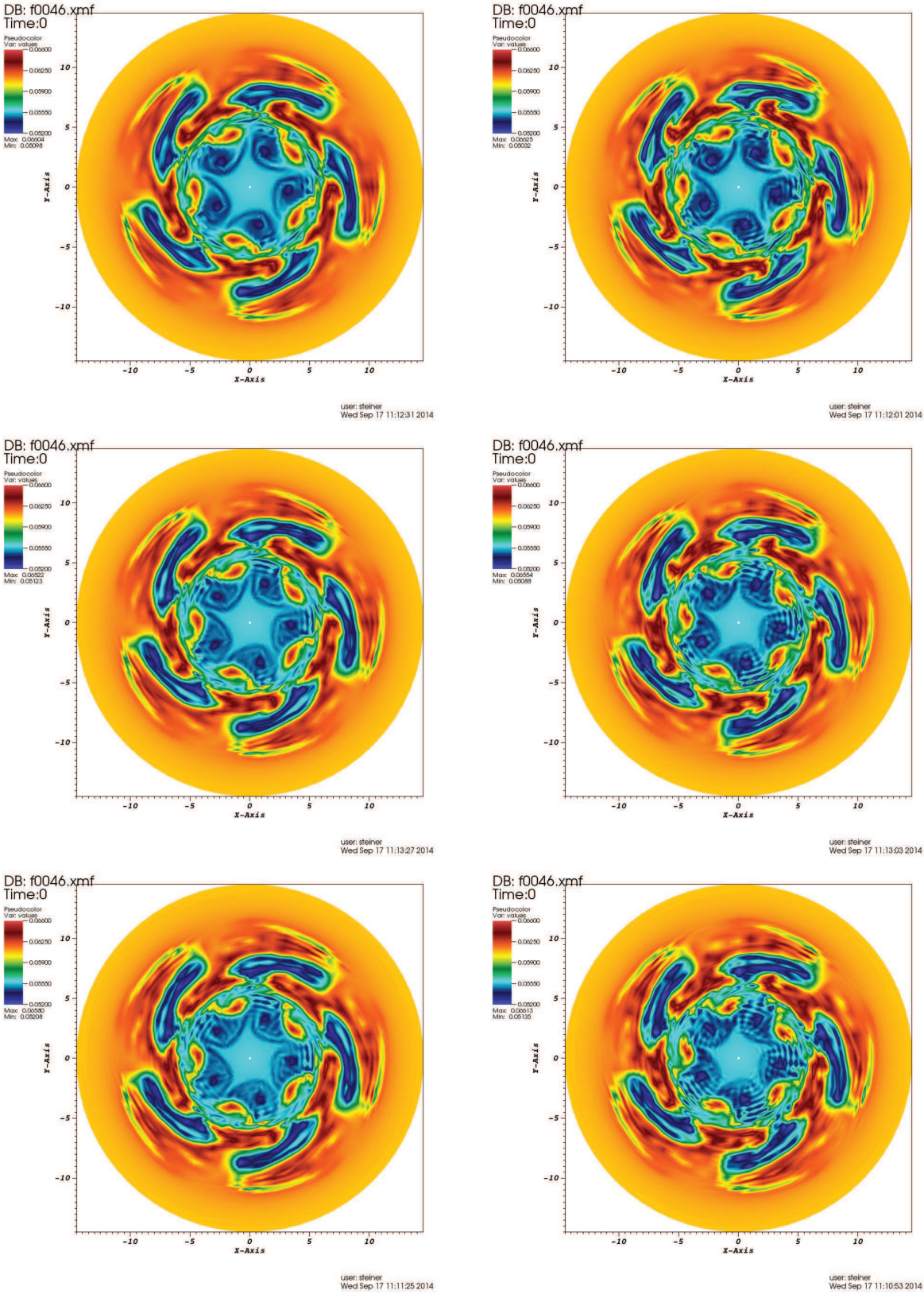


FIGURE 7.2 – Coupe poloïdale $f(r, \theta, z = 0, v_{\|} = 0)$ pour $64 \times 64 \times 32 \times 64$, $T = 4600$ avec $\Delta t = 4$ (à gauche) et $\Delta t = 2$ (à droite) ; PPM1 (en haut), PPM2 (au milieu) et splines cubiques (en bas).

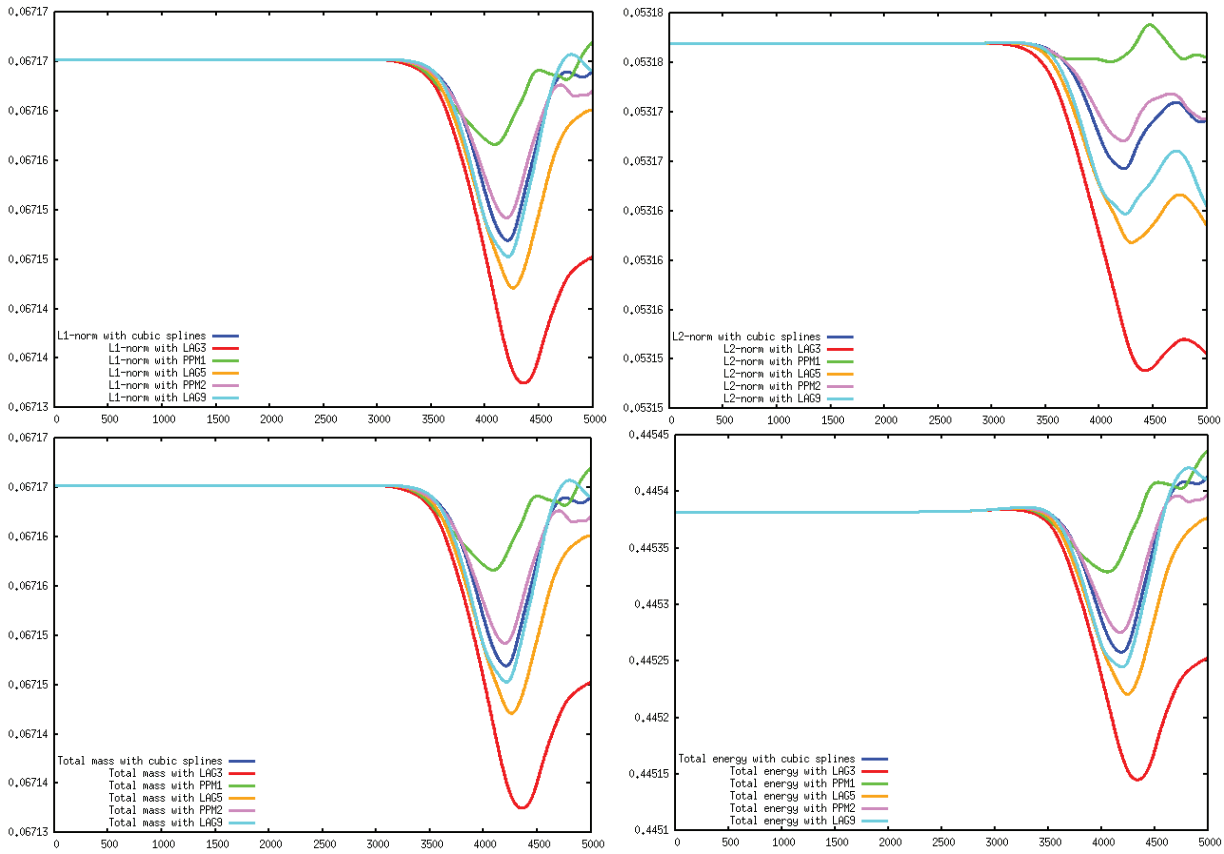


FIGURE 7.3 – Evolution en temps de la norme L^1 (en haut à gauche), de la norme L^2 (en haut à droite), de la masse totale (en bas à gauche) et de l'énergie totale (en bas à droite) avec méthode par splines cubiques (en bleu), LAG3 (en rouge), PPM1 (en vert), LAG5 (en orange), PPM2 (en violet) et LAG9 (en cyan). Paramètres : $64 \times 64 \times 32 \times 64$, $\Delta t = 2$, temps final : $T = 5000$.

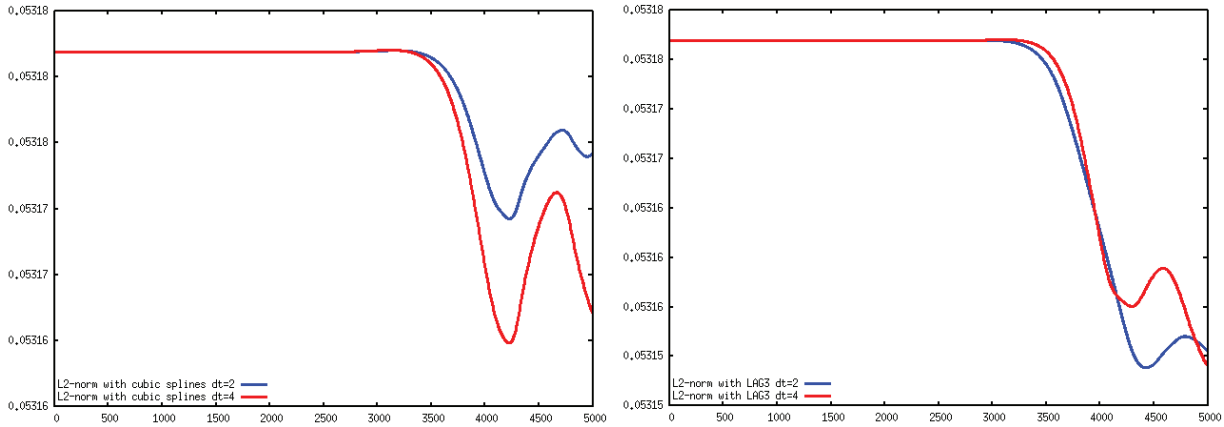


FIGURE 7.4 – Evolution en temps de la norme L^2 avec méthode par splines cubiques (à gauche) et LAG3 (à droite) pour $\Delta t = 4$ (en rouge) et $\Delta t = 2$ (en bleu). Paramètres : $64 \times 64 \times 32 \times 64$, temps final : $T = 5000$.

Δt	LAG 3	PPM 1	LAGH 5	PPM 2	LAGH 9	Splines cubiques
4	119	127	130	137	140	115
2	251	251	251	258	262	228

TABLE 7.1 – Temps (en min.) pour atteindre le temps final $T = 5000$, sur le calculateur irma-hpc2 avec 16 processeurs, $N_r \times N_\theta \times N_z \times N_v = 64 \times 64 \times 32 \times 64$, $\Delta t = 2$ ou 4.

Chapitre 8

Gyromoyenne

Ce chapitre concerne le calcul numérique de l'opérateur de gyromoyenne pour un maillage polaire qui est le cadre du code gyrocinétique GYSELA [80, 90]. Nous suivons un travail antérieur en géométrie cartésienne [87]. Nous proposons une alternative à l'approximation de Padé classique qui est utilisée dans GYSELA et qui est connue pour n'être valable que pour de faibles rayons de Larmor. La méthode est basée sur une intégration directe et une interpolation. Proche de la méthode déjà utilisée dans le code GENE (voir [92, 91]), elle est appliquée à des simulations gyrocinétiques.

8.1 Définition de l'opérateur de gyromoyenne

Soit $\boldsymbol{\rho}$ le gyro-rayon qui est transverse à $\mathbf{b} = \mathbf{B}/B$ (où $\mathbf{B}$ est le champ magnétique) et qui dépend de la gyrophase $\alpha \in [0, 2\pi]$, i.e

$$\boldsymbol{\rho} = \rho(\cos(\alpha)\mathbf{e}_{\perp 1} + \sin(\alpha)\mathbf{e}_{\perp 2})$$

Ici $\mathbf{e}_{\perp 1}$ et $\mathbf{e}_{\perp 2}$ sont les vecteurs unitaires d'une base cartésienne dans le plan perpendiculaire à la direction du champ magnétique $\mathbf{b}$. Soit $\mathbf{x}_G$ les coordonnées radiales du centre-guide et $\mathbf{x}$ la position des particules dans l'espace réel. Ces deux quantités diffèrent par le rayon de Larmor $\boldsymbol{\rho}$, i.e $\mathbf{x} = \mathbf{x}_G + \boldsymbol{\rho}$.
Soit $f : (r, \theta) \in \mathbb{R}^+ \times \mathbb{R} \mapsto f(r, \theta)$ une fonction polaire et $g : (x_1, x_2) \in \mathbb{R}^2 \mapsto g(x_1, x_2)$ la fonction définie par $g(r\cos(\theta), r\sin(\theta)) = f(r, \theta)$ pour tout (r, θ). La fonction f (resp. g) représente les quantités de champ définies en $\mathbf{x}$ en coordonnées polaires (resp. cartésiennes). La gyromoyenne $\mathcal{J}_\rho(f)$ de f dépendant des coordonnées spatiales est définie par

$$\mathcal{J}_\rho(f)(r, \theta) = \frac{1}{2\pi}\int_0^{2\pi} g(\mathbf{x}_G + \boldsymbol{\rho})d\alpha.$$

où $\mathbf{x}_G = r(\cos(\theta), \sin(\theta))$. Ce processus de gyromoyenne consiste à calculer une moyenne sur le cercle de Larmor. Elle tend à amortir toute fluctuation qui se développe à une échelle inférieure au rayon de Larmor.
En introduisant $\widehat{f}(\mathbf{k})$ la transformée de Fourier de f, avec $\mathbf{k} = k(\cos(\theta), \sin(\theta))$ le vecteur

d'onde, l'opération de gyromoyenne se lit

$$\begin{aligned}\mathcal{J}_\rho(f)(\mathbf{x}_G) &= \int_0^{2\pi}\frac{d\alpha}{2\pi}\int_{\mathbb{R}^2}\frac{d^2\mathbf{k}}{(2\pi)^3}\,\widehat{f}(\mathbf{k})\exp\{i\mathbf{k}\cdot(\mathbf{x}_G+\boldsymbol{\rho})\}\\ &= \int_{\mathbb{R}^2}\frac{d^2\mathbf{k}}{(2\pi)^3}\left[\int_0^{2\pi}\frac{d\alpha}{2\pi}\exp(ik_\perp\rho\cos\alpha)\right]\times\\ &\qquad \widehat{f}(\mathbf{k})\exp(i\mathbf{k}\cdot\mathbf{x}_G)\end{aligned}$$

où $k_\perp$ est la norme de la composante transversale au vecteur d'onde $\mathbf{k}_\perp = \mathbf{k} - (\mathbf{b}.\mathbf{k})\mathbf{b}$. Soit n un entier et considérons J_n la fonction de Bessel de première espèce et d'ordre n, i.e $\forall z\in\mathbb{C}$, $J_n(z) = \frac{i^{-n}}{\pi}\int_0^\pi \exp(iz\cos\theta)\cos(n\theta)\,d\theta$. Par conséquent, l'opération de gyromoyenne précédente peut être exprimée en fonction de la fonction de Bessel du premier ordre J_0 par

$$\mathcal{J}_\rho(f)(\mathbf{x}_G) = \int_{-\infty}^{+\infty}\frac{d^3\mathbf{k}}{(2\pi)^3}\,J_0(k_\perp\rho)\widehat{f}(\mathbf{k})\mathrm{e}^{\mathrm{i}\mathbf{k}\cdot\mathbf{x}_\mathrm{G}}. \tag{8.1.1}$$

Considérons un maillage polaire uniforme $(r,\theta)\in[r_{\min},r_{\max}]\times[0,2\pi[$ avec $N_r\times N_\theta$ cellules, notre objectif est d'approximer l'opérateur

$$(f_{j,k})\in\mathbb{R}^{(N_r+1)\times N_\theta}\mapsto(\mathcal{J}_\rho(f)_{j,k})\in\mathbb{R}^{(N_r+1)\times N_\theta}.$$

Considérant l'expression (8.1.1) dans l'espace de Fourier, la gyromoyenne se réduit à une multiplication par la fonction de Bessel d'argument $k_\perp\rho$. En effet, la transformée de Fourier de $\mathcal{J}_\rho(f)$ peut être écrite comme

$$\begin{aligned}\widehat{\mathcal{J}_\rho(f)}(\mathbf{k}) &= \frac{1}{2\pi}\int_{\mathbb{R}^2}\int_0^{2\pi} f(\mathbf{x}_G+\boldsymbol{\rho})d\alpha\; e^{-i\mathbf{x}_G\cdot\mathbf{k}}d\mathbf{x}_G\\ &= \frac{1}{2\pi}\int_0^{2\pi}\int_{\mathbb{R}^2} f(\mathbf{x}_G+\boldsymbol{\rho})e^{-i(\mathbf{x}_G+\boldsymbol{\rho})\cdot\mathbf{k}}d\mathbf{x}_G\; e^{i\boldsymbol{\rho}\cdot\mathbf{k}}d\alpha\\ &= \left(\frac{1}{2\pi}\int_0^{2\pi} e^{ik\rho\cos(\alpha-\theta)}d\alpha\right)\widehat{f}(\mathbf{k})\end{aligned}$$

ce qui conduit à

$$\widehat{\mathcal{J}_\rho(f)}(\mathbf{k}) = J_0(k\rho)\widehat{f}(\mathbf{k}). \tag{8.1.2}$$

Cette opération est immédiate en géométrie simple avec des conditions aux bords périodiques, ainsi que dans les codes locaux. Inversement, dans le cas de codes globaux, l'utilisation de la transformée de Fourier n'est pas applicable pour deux raisons principales : (i) les conditions aux bords radiales ne sont pas périodiques, et (ii) la dépendance radiale du rayon de Larmor doit être prise en compte. Plusieurs approches ont été développées pour surmonter cette difficulté. La méthode la plus répandue pour ce processus de gyromoyennisation est d'utiliser une formule de quadrature. Dans ce contexte, l'intégrale sur le cercle de giration est généralement approchée par une somme de quatre points sur ce cercle [96]. C'est rigoureusement équivalent à considérer le développement de Taylor de la fonction de Bessel à l'ordre 2, à savoir $J_0(k_\perp\rho)\simeq 1-(k_\perp\rho)^2/4$, et équivalent à calculer le Laplacien transverse au second ordre en utilisant des différences finies. Cette méthode a été étendue pour s'adapter à de grands rayons de Larmor [93], c'est-à-dire que le nombre de points (à partir de quatre)

augmente de façon linéaire avec le rayon de Larmor afin de garantir un nombre de points par longueur d'arc constant sur le cercle d'intégration. Dans cette approche – utilisée par exemple dans [94] and [95] – les points, qui sont distribués de manière équidistante sur le cercle, sont décalés pour chaque particule (ou marqueur) par un angle aléatoire calculé à chaque pas de temps. Ceci est effectué avec un formalisme d'éléments finis et permet donc une précision d'ordre élevé en mettant le problème sous forme matricielle .
Dans [87], l'influence de l'opérateur d'interpolation (qui est d'une grande importance lorsque les points de quadrature ne coïncident pas avec les points de la grille) a été étudiée et a montré que les splines cubiques sont un bon candidat. Certaines techniques utilisées dans [87] utilisent le fait d'être en coordonnées cartésiennes et ne sont plus valables en géométrie polaire. Dans ce chapitre, nous présentons une méthode basée sur une intégration directe de l'opérateur de gyromoyenne qui est directement applicable pour un code gyrocinétique global en géométrie toroïdale comme par exemple le code GYSELA. Cette nouvelle approche est testée pour deux méthodes d'interpolation différentes, l'une basée sur les splines cubiques et l'autre sur les polynômes d'Hermite. Les deux sont comparées à l'approximation de Padé.

8.2 Méthode basée sur l'approximation de Padé

Une solution possible pour évaluer l'opérateur de gyromoyenne est d'approximer la fonction de Bessel avec son développement de Padé : $J_{\text{Pade}}(k\rho) = 1/\left[1+(k\rho)^2/4\right]$ (voir par exemple [97]). Comme décrit dans ce qui suit, une telle approximation de Padé nécessite l'inversion de l'opérateur Laplacien dans l'espace réel. En effet, en utilisant cette approximation, la relation (8.1.2) se lit

$$\left(1+\frac{(k\rho)^2}{4}\right)\widehat{\mathcal{J}_\rho(f)}(\mathbf{k}) = \widehat{f}(\mathbf{k})$$

ce qui correspond dans l'espace réel à

$$\left(1-\frac{\rho^2}{4}\Delta\right)\mathcal{J}_\rho(f)(\mathbf{x}) = f(\mathbf{x}).$$

Nous projetons f et $J(f)$ dans la base de Fourier :

$$f(r,\theta) \approx \sum_{n=0}^{N_\theta-1} A_n(r)e^{in\theta}, \qquad \mathcal{J}_\rho(f)(r,\theta) \approx \sum_{n=0}^{N_\theta-1} B_n(r)e^{in\theta}.$$

Le laplacien en coordonnées polaires s'exprime par

$$\Delta = \frac{\partial^2}{\partial r^2} + \frac{1}{r}\frac{\partial}{\partial r} + \frac{1}{r^2}\frac{\partial^2}{\partial\theta^2}$$

et nous obtenons pour $n = 0...N_\theta - 1$

$$-\frac{\rho^2}{4}B_n''(r) - \frac{\rho^2}{4r}B_n'(r) + \left(1+\frac{\rho^2 n^2}{4r^2}\right)B_n(r) \;=\; A_n(r)$$

qui peut être résolu par différences finies. Pour les résultats présentés dans la suite, nous utilisons les différences finies d'ordre 2 ce qui conduit à un système tridiagonal.

Remarque 8.2.1. *Dans le cas d'un ρ non constant, nous avons également testé l'opérateur $1 - \frac{\rho}{4}\nabla \cdot (\rho\nabla)$; nous avons constaté que l'opérateur $1 - \frac{\rho^2}{4}\Delta$ donne de meilleurs résultats.*

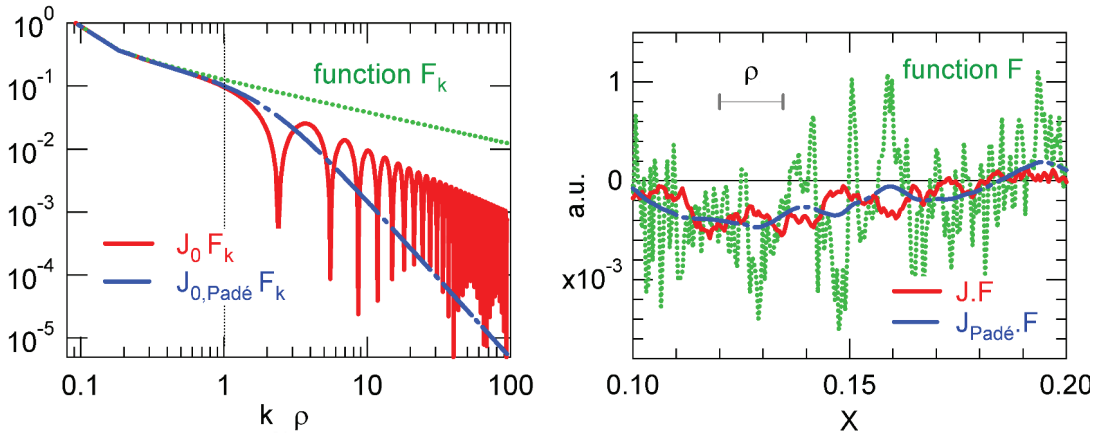
Cette approximation de Padé donne la bonne limite dans le cas limite des grandes longueurs d'onde $k\rho \ll 1$, tout en gardant J_{Pade} fini pour la limite opposée $k\rho \to \infty$. L'inconvénient est un sur-armortissement des petites échelles : dans la limite des grands arguments $x \to \infty$, $J_{\text{Pade}}(x) \to 4/x^2$, tandis que $J_0(x) \to (2/\pi x)^{1/2}\cos(x - \pi/4)$ (voir Fig. 8.2). La Fig. 8.2b présente, dans l'espace réel, la gyromoyenne exacte d'une fonction aléatoire (en rouge) et son approximation de Padé (en bleu). La courbe bleue est proche de la solution exacte mais ne la reproduit pas exactement.

La méthode proposée dans la section suivante, qui n'est plus basée sur une approximation de la fonction de Bessel, mais sur le calcul direct de l'intégrale sur un cercle de rayon ρ, a été mise au point pour remédier à cet inconvénient.

FIGURE 8.1 – La fonction de Bessel $J_0(k\rho)$ comparée à son approximation de Padé $1/\left[1 + (k\rho)^2/4\right]$.

8.3 Méthode basée sur l'interpolation

Dans cette section, nous décrivons le calcul de l'opérateur de gyromoyenne dans l'espace réel. Cette méthode implique essentiellement des interpolations sur le cercle de Larmor. Nous plaçons N points uniformément répartis sur le cercle d'intégration et nous approchons la valeur de la fonction en ces points par interpolation. La gyromoyenne est alors obtenue



(a) (b)

FIGURE 8.2 – Opérateurs de gyromoyenne exact et approché appliqués à une fonction arbitraire F_k présentant un large spectre allant de faibles à de grandes longueurs d'onde par rapport au rayon de Larmor ρ : (a) Représentation dans l'espace de Fourier, (b) Représentation dans l'espace réel (figures de [97]).

par la formule de quadrature des rectangles en ces points. Plus précisément, pour un point (r_j, θ_k) donné, la gyromoyenne en ce point est approximée par

$$J_\rho(f)_{j,k} \simeq \frac{1}{2\pi} \sum_{\ell=0}^{N-1} \mathcal{P}(f)(r_j \cos\theta_k + \rho\cos\alpha_\ell, r_j \sin\theta_k + \rho \sin\alpha_\ell)\Delta\alpha,$$

où $\alpha_\ell = \ell\Delta\alpha$ et $\Delta\alpha = 2\pi/N$. Etant donné que les points de quadrature ne coïncident pas avec des points de la grille, nous introduisons un opérateur d'interpolation $\mathcal{P}$ qui peut être

— l'interpolation d'Hermite,

— l'interpolation par splines cubiques.

Lorsque les points de la somme précédente sont en dehors du domaine, nous faisons une projection radiale sur le bord du domaine. Plus précisément,

— si $r < r_{\min}$ alors $\mathcal{P}(f)(r,\theta)$ sera remplacé par $\mathcal{P}(f)(r_{\min},\theta)$

— si $r > r_{\max}$ alors $\mathcal{P}(f)(r,\theta)$ sera remplacé par $\mathcal{P}(f)(r_{\max},\theta)$.

Remarque 8.3.1. *Dans les applications aux simulations gyrocinétiques, l'opérateur de gyromoyenne est appliqué à $f - f_{eq}$ où f est la fonction de distribution et f_{eq} est la fonction de distribution à l'équilibre. La fonction $f - f_{eq}$ a des valeurs proches de 0 aux bords du domaine.*

Comme détaillé dans [75], l'interpolation peut être reformulée en un produit matrice-vecteur

$$\mathcal{P}(f)(r_j \cos\theta_k + \rho\cos\alpha_\ell, r_j \sin\theta_k + \rho\sin\alpha_\ell) = (A_\ell c)_{j,k},$$

où c correspond aux coefficients de splines (interpolation par splines cubiques) ou les valeurs de la fonction (cas Hermite) de telle sorte à ce que la gyromoyenne peut être elle-même considérée comme un produit matrice-vecteur :

$$J_\rho(f)_{j,k} = \frac{1}{2\pi} \sum_{\ell=0}^{N-1} (A_\ell f)_{j,k}\Delta\alpha = (A_\rho c)_{j,k}.$$

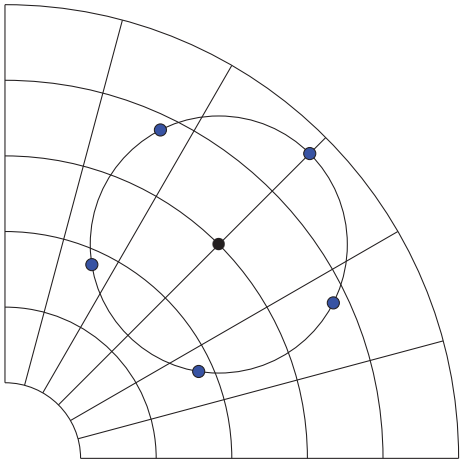


FIGURE 8.3 – Configuration spatiale de la méthode basée sur l'interpolation.

En conséquence, pour un rayon de Larmor donné ρ, la matrice A_ρ peut être stockée une fois pour toutes. Pour chaque méthode, 2 versions sont implémentées :

— une version basique

— une version avec précalcul où nous calculons d'abord la matrice A_ρ telle que

$$(\mathcal{J}_\rho(f)_{j,k}) = A_\rho c,$$

où c sont les coefficients de splines (vecteur de taille $(N_r+1)N_\theta$) ou les valeurs de la fonction et de ses dérivées dans le cas de l'interpolation d'Hermite (la taille vaut alors $4(N_r+1)N_\theta$).

Remarque 8.3.2. *Notons que pour l'interpolation d'Hermite, nous devons d'abord calculer les dérivées à chaque interface entre les cellules. Ces dérivées sont reconstruites par différences finies centrées d'ordre pair arbitraire (on peut aussi utiliser de l'ordre impair en ayant seulement une reconstruction C^0, mais alors la taille de c passerait à $9(N_r+1)N_\theta$). Dans les résultats numériques, nous prendrons l'ordre 4. Les temps d'exécution pour calculer la gyromoyenne avec différentes méthodes d'interpolation sont donnés dans le tableau 8.1. Nous observons que dans le cas de l'interpolation d'Hermite, l'ordre n'impacte que peu les performances.*

Remarque 8.3.3. *Les comparaisons de temps sont données dans les Tables 8.1 et 8.2. L'utilisation de la version avec précalcul est ici assez efficace, comme le rayon de Larmor ρ est fixe et la matrice est la même pour chaque valeur de θ, ce qui implique que le stockage est réduit ; mais la version de base permet de donner une indication approximative du temps qui serait utilisé pour des situations plus générales (où par exemple le stockage serait un problème) et qui ne sont pas prises en considération pour le moment.*

Remarque 8.3.4. *On peut s'interroger sur le coût numérique de la méthode d'Hermite par rapport à l'approximation de Padé, surtout pour un grand rayon. Heureusement, la situation change dans les applications qui sont au moins en dimension 4, tandis que la gyromoyenne est appliquée uniquement en $3D$. A titre d'exemple, nous avons obtenu les temps suivants dans le cas d'une simulation drift kinetic (voir sous-section 8.5) : $42s$. pour PADE, $43s$.*

ρ	Hermite (4)	(6)	(10)	(18)	splines	Padé
0	6	7	9	14	8	0.6
0.001	10	11	13	18	12	2
0.01	40	41	43	48	53	2
0.1	446	442	448	453	594	2

TABLE 8.1 – Temps (en s.) de Hermite précalcul et Padé en fonction de ρ avec différents ordres (4, 6, 10 or 18) pour la reconstruction des dérivées dans le cas de l'interpolation d'Hermite (voir la remarque 8.3.2). Paramètres : $r_{\min} = 0.1, r_{\max} = 0.9, N_r = N_\theta = 512, N = 1024$, 100 itérations de la gyromoyenne.

ρ	Hermite basique	Hermite précalcul
0	301	0.6
0.001	316	0.8
0.01	312	1
0.1	304	5

TABLE 8.2 – Temps (en s.) en fonction de ρ pour Hermite sans et avec précalcul. Paramètres : $r_{\min} = 0.1, r_{\max} = 0.9, N_r = N_\theta = 128, N = 1024$, ordre d'interpolation : 4, 100 itérations de la gyromoyenne.

pour l'interpolation d'Hermite avec précalcul (en utilisant 1024 *points de quadrature). Sans précalcul, le temps pour Hermite est de* 47s. *avec* 16 *points de quadrature et* 270s. *avec* 1024 *points de quadrature. Les calculs sont effectués sur un cluster local de l'Université de Strasbourg en utilisant 16 processeurs (grille de* $32 \times 32 \times 32 \times 64$ *points,* 100 *itérations). D'autres mesures de temps seront détaillées dans le paragraphe 8.5.*

Remarque 8.3.5. *Cette méthode, qui se réduit à une simple interpolation, est très souple, car elle ne dépend pas du maillage. Elle peut également être facilement étendue à d'autres systèmes de coordonnées.*

8.4 Comparaison numérique avec des solutions analytiques

8.4.1 Définition d'une classe de solutions analytiques dépendant des conditions aux bords

Tout d'abord, nous donnons une famille de fonctions dont la gyromoyenne est analytiquement connue. Pour ces fonctions, on obtient la gyromoyenne simplement en les multipliant par la fonction de Bessel.
Soit $m \geq 0$ un entier et C_m la fonction de Bessel de première espèce (notée par J_m) ou

la fonction de Bessel de seconde espèce (notée par Y_m), voir [98]. La proposition suivante donne l'expression analytique de la gyromoyenne des fonctions de type Fourier-Bessel.

Proposition. Soit $z \in \mathbb{C}$. La gyromoyenne de

$$f(r,\theta) = C_m(zr)e^{im\theta}$$

se lit

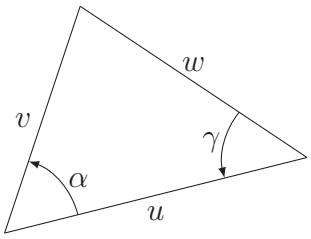
$$\mathcal{J}_\rho(f)(r_0,\theta_0) = J_0(z\rho)C_m(zr_0)e^{im\theta_0}.$$

Preuve : Par définition,

$$\begin{aligned}&\mathcal{J}_\rho(f)(r,\theta)\\ &= \frac{1}{2\pi}\int_0^{2\pi}\int_0^{+\infty}\int_0^{2\pi} C_m(zr_0)e^{im\theta_0}\delta_{\{\vec{x_0}=\vec{x}+\vec{\rho}\}}d\alpha dr_0 d\theta_0\end{aligned}$$

où $\vec{x_0} = r_0(\cos(\theta_0),\sin(\theta_0))$, $\vec{x} = r_0(\cos(\theta),\sin(\theta))$ et $\vec{\rho} = \rho(\cos(\alpha),\sin(\alpha))$.

Le théorème d'additivité de Graf pour les fonctions de Bessel (voir [83]) affirme que si u, v et w sont les longueurs d'un triangle et α, γ les angles comme indiqué dans la figure suivante :



alors pour tout entier m et pour tout nombre complexe z,

$$C_m(zw)e^{im\gamma} = \sum_{k=-\infty}^{\infty} C_{m+k}(zu)J_k(zv)e^{ik\alpha}.$$

Ainsi, on obtient, avec $v = \rho$, $w = r_0$, $u = r$, $\gamma = \theta_0 - \theta$ et $\alpha = \alpha$,

$$\begin{aligned}
&\mathcal{J}_\rho(f)(r,\theta) \\
&= e^{im\theta}\frac{1}{2\pi}\int_0^{2\pi}\int_0^{+\infty}\int_0^{2\pi} C_m(zr_0)e^{im(\theta_0-\theta)} \times \\
&\quad \delta_{\{\vec{x_0}=\vec{x}+\vec{\rho}\}} d\alpha dr_0 d\theta_0 \\
&= e^{im\theta}\frac{1}{2\pi}\int_0^{2\pi}\int_0^{+\infty}\int_0^{2\pi}\left(\sum_{k=-\infty}^{\infty} C_{m+k}(zr)J_k(z\rho)e^{ik\alpha}\right) \times \\
&\quad \delta_{\{\vec{x_0}=\vec{x}+\vec{\rho}\}} d\alpha dr_0 d\theta_0 \\
&= e^{im\theta}\left(\sum_{k=-\infty}^{\infty} C_{m+k}(zr)J_k(z\rho)\right) \times \\
&\quad \frac{1}{2\pi}\int_0^{2\pi}\int_0^{+\infty}\int_0^{2\pi} e^{ik\alpha}\delta_{\{\vec{x_0}=\vec{x}+\vec{\rho}\}} d\alpha dr_0 d\theta_0 \\
&= e^{im\theta}\left(\sum_{k=-\infty}^{\infty} C_{m+k}(zr)J_k(z\rho)\right) \times \frac{1}{2\pi}\int_0^{2\pi} e^{ik\alpha} d\alpha.
\end{aligned}$$

Nous utilisons le fait que

$$\frac{1}{2\pi}\int_0^{2\pi} e^{ik\alpha} d\alpha = \delta_{k,0}$$

pour conclure que

$$\mathcal{J}_\rho(f)(r,\theta) = J_0(z\rho)C_m(zr)e^{im\theta}.$$

□

Dans ce qui suit, nous donnons quelques exemples de ces fonctions test en fonction des conditions aux bords que l'on souhaite tester.

Exemples

1. $r_{\min} = 0$ *et des conditions de Dirichlet homogènes en* $r_{\max}$.

   Ici nous considérons un disque $[0, r_{\max}] \times [0, 2\pi]$ et la fonction

   $$f_1(r,\theta) = J_m\left(r\frac{j_{m,\ell}}{r_{\max}}\right)e^{im\theta}$$

   où $j_{m,\ell}$ est le $\ell^{ème}$ zéro de J_m. La fonction f_1 vérifie la condition de Dirichlet :

   $$f_1(r_{\max},\theta) = 0,\ 0 \le \theta < 2\pi,$$

   et sa gyromoyenne vaut

   $$\mathcal{J}_\rho(f_1)(r_0,\theta_0) = J_0\left(\rho\frac{j_{m,\ell}}{r_{\max}}\right) f_1(r_0,\theta_0).$$

2. *Conditions de Dirichlet homogènes en* $r_{\min} > 0$ *et* $r_{\max}$.

   La fonction suivante est définie sur l'anneau $[r_{\min}, r_{\max}] \times [0, 2\pi]$:

$$f_2(r,\theta) = \left(J_m(\gamma_{m,\ell}) Y_m\left(r\frac{\gamma_{m,\ell}}{r_{\max}}\right) - Y_m(\gamma_{m,\ell}) J_m\left(r\frac{\gamma_{m,\ell}}{r_{\max}}\right)\right) e^{im\theta}$$

   où $\gamma_{m,\ell}$ est le $\ell^{ème}$ zéro de

$$y \mapsto J_m(y) Y_m\left(y\frac{r_{\min}}{r_{\max}}\right) - Y_m(y) J_m\left(y\frac{r_{\min}}{r_{\max}}\right).$$

   La fonction f_2 vérifie les conditions de Dirichlet :

$$f_2(r_{\min},\theta) = 0, \quad f_2(r_{\max},\theta) = 0,\ 0 \le \theta < 2\pi,$$

   et sa gyromoyenne vaut

$$\mathcal{J}_\rho(f_2)(r_0,\theta_0) = J_0\left(\rho\frac{\gamma_{m,\ell}}{r_{\max}}\right) f_2(r_0,\theta_0).$$

3. *Conditions de Neumann homogènes en* $r_{\min} > 0$ *and* $r_{\max}$.

   La fonction suivante est définie sur l'anneau $[r_{\min}, r_{\max}] \times [0, 2\pi]$:

$$f_3(r,\theta) = \left(J'_m(\eta_{m,\ell}) Y_m\left(r\frac{\eta_{m,\ell}}{r_{\max}}\right) - Y'_m(\eta_{m,\ell}) J_m\left(r\frac{\eta_{m,\ell}}{r_{\max}}\right)\right) e^{im\theta}$$

   où $\eta_{m,\ell}$ est le $\ell^{ème}$ zéro de

$$y \mapsto J'_m(y) Y'_m\left(y\frac{r_{\min}}{r_{\max}}\right) - Y'_m(y) J'_m\left(y\frac{r_{\min}}{r_{\max}}\right).$$

   La fonction f_3 vérifie les conditions de Neumann :

$$\partial_r f_3(r_{\min},\theta) = 0, \quad \partial_r f_3(r_{\max},\theta) = 0,\ 0 \le \theta < 2\pi,$$

   et sa gyromoyenne vaut

$$\mathcal{J}_\rho(f_3)(r_0,\theta_0) = J_0\left(\rho\frac{\eta_{m,\ell}}{r_{\max}}\right) f_3(r_0,\theta_0).$$

   Nous avons ici utilisé le fait que pour $\mathcal{C}_n = J_n$ ou Y_n, la dérivée satisfait la relation :

$$\mathcal{C}'_n(r) = -\mathcal{C}_{n+1}(r) + \frac{n\mathcal{C}_n(r)}{r}.$$

Nous montrons dans la Fig. 8.4 les parties réelle et imaginaire de la fonction

$$(r,\theta) \in [0,5] \times [0,2\pi] \mapsto J_1(r) e^{i\theta}.$$

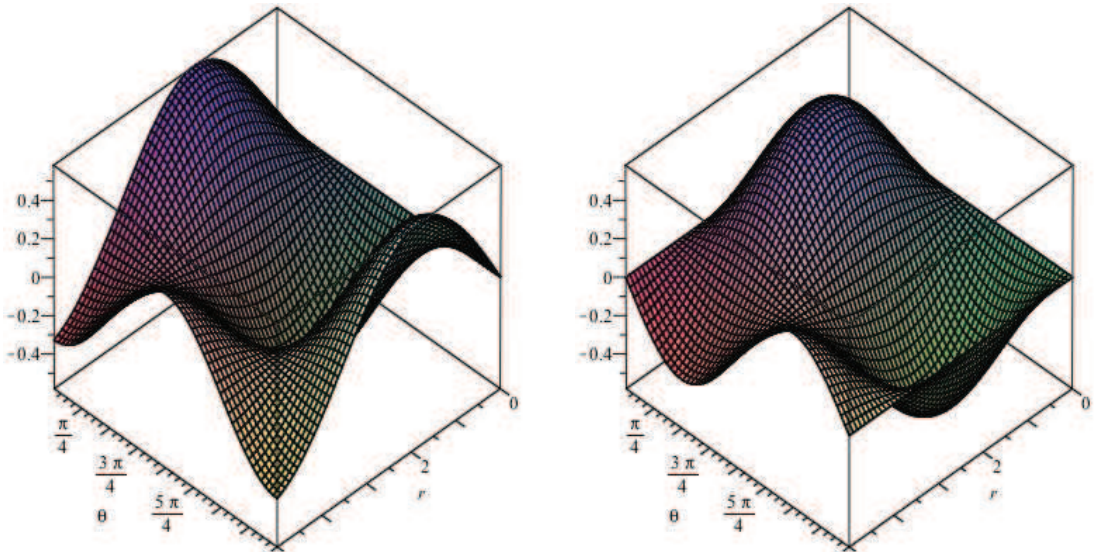


FIGURE 8.4 – Parties réelle et imaginaire de la fonction $(r, \theta) \mapsto J_1(r)\exp(i\theta)$.

ρ $[\rho/r_{\max}]$	Padé	$N = 4$	8	16	1024
0 [0]	10^{-17}	10^{-17}	10^{-17}	10^{-17}	10^{-17}
10^{-3} [0.001]	4.10^{-11}	3.10^{-6}	3.10^{-6}	3.10^{-6}	3.10^{-6}
10^{-2} [0.011]	2.10^{-8}	3.10^{-6}	1.10^{-6}	4.10^{-7}	5.10^{-7}
10^{-1} [0.111]	2.10^{-4}	3.10^{-4}	1.10^{-5}	4.10^{-8}	5.10^{-8}

TABLE 8.3 – Comparaison entre Padé et Hermite ($m = 1$).

8.4.2 Résultats numériques

Dans cette partie, les différentes méthodes numériques sont comparées dans le cas du second cas test (Conditions de Dirichlet homogènes en $r_{\min}$ and $r_{\max}$) avec $r_{\min} = 0.1$, $r_{\max} = 0.9$, $\ell = 1$ et $m = 1, 5, 20, 60$. Nous considérons $N_r = N_\theta = 512$ dans les Tables 8.3 – 8.8 et $N_r = N_\theta = 1024$ dans les Tables 8.9 – 8.10. Dans les Tables 8.3 et 8.4, nous donnons l'erreur en norme L^2 pour la fonction gyromoyennée avec $m = 1$ tandis que les Tables 8.5, 8.6 réfèrent à $m = 5$, les Tables 8.7, 8.8 réfèrent à $m = 20$ et les Tables 8.9, 8.10 réfèrent à $m = 60$. Dans la Table 8.11, nous utilisons divers ordres d'interpolation pour Hermite. Notons que pour chaque méthode, l'erreur est calculée sur le domaine $[r_{\min} + \rho, r_{\max} - \rho]$.

Remarque 8.4.1.

1. *Pour $\rho = 0$, toutes les méthodes sont exactes.*
2. *La méthode basée sur l'interpolation donne à peu près les mêmes résultats qu'avec l'interpolation d'Hermite ou l'interpolation par splines cubiques.*
3. *La méthode basée sur l'approximation de Padé donne de très bons résultats pour les petites valeurs de ρ.*
4. *Pour de grandes valeurs de ρ, la méthode basée sur l'interpolation donne de meilleurs*

ρ $[\rho/r_{\max}]$	Padé	$N=4$	8	16	1024
0 [0]	10^{-17}	10^{-17}	10^{-17}	10^{-17}	10^{-15}
10^{-3} [0.001]	4.10^{-11}	3.10^{-6}	3.10^{-6}	3.10^{-6}	3.10^{-6}
10^{-2} [0.011]	2.10^{-8}	3.10^{-6}	1.10^{-6}	5.10^{-7}	6.10^{-7}
10^{-1} [0.111]	2.10^{-4}	3.10^{-4}	1.10^{-5}	4.10^{-8}	9.10^{-8}

TABLE 8.4 – Comparaison entre Padé et splines ($m = 1$).

ρ $[\rho/r_{\max}]$	Padé	$N=4$	8	16	1024
0 [0]	10^{-17}	10^{-17}	10^{-17}	10^{-17}	10^{-17}
10^{-3} [0.001]	1.10^{-8}	5.10^{-6}	5.10^{-6}	5.10^{-6}	5.10^{-6}
10^{-2} [0.011]	2.10^{-6}	2.10^{-6}	1.10^{-6}	8.10^{-7}	7.10^{-7}
10^{-1} [0.111]	1.10^{-3}	1.10^{-3}	3.10^{-5}	1.10^{-7}	6.10^{-8}

TABLE 8.5 – Comparaison entre Padé et Hermite ($m = 5$).

ρ $[\rho/r_{\max}]$	Padé	$N=4$	8	16	1024
0 [0]	10^{-17}	10^{-17}	10^{-17}	10^{-17}	10^{-17}
10^{-3} [0.001]	1.10^{-8}	5.10^{-6}	5.10^{-6}	5.10^{-6}	5.10^{-6}
10^{-2} [0.011]	2.10^{-6}	2.10^{-6}	1.10^{-6}	8.10^{-7}	8.10^{-7}
10^{-1} [0.111]	1.10^{-3}	1.10^{-3}	3.10^{-5}	2.10^{-7}	1.10^{-7}

TABLE 8.6 – Comparaison entre Padé et splines ($m = 5$).

ρ $[\rho/r_{\max}]$	Padé	$N=4$	8	16	1024
0 [0]	10^{-18}	10^{-17}	10^{-17}	10^{-17}	10^{-17}
10^{-3} [0.001]	1.10^{-8}	5.10^{-6}	4.10^{-6}	4.10^{-6}	4.10^{-6}
10^{-2} [0.011]	6.10^{-6}	1.10^{-6}	1.10^{-6}	7.10^{-7}	7.10^{-7}
10^{-1} [0.111]	9.10^{-3}	3.10^{-3}	1.10^{-5}	9.10^{-8}	6.10^{-8}

TABLE 8.7 – Comparaison entre Padé et Hermite ($m = 20$).

ρ $[\rho/r_{\max}]$	Padé	$N=4$	8	16	1024
0 [0]	10^{-18}	10^{-17}	10^{-17}	10^{-17}	10^{-16}
10^{-3} [0.001]	1.10^{-8}	4.10^{-6}	4.10^{-6}	4.10^{-6}	4.10^{-6}
10^{-2} [0.011]	6.10^{-6}	1.10^{-6}	1.10^{-6}	8.10^{-7}	8.10^{-7}
10^{-1} [0.111]	9.10^{-3}	3.10^{-3}	1.10^{-5}	1.10^{-7}	1.10^{-7}

TABLE 8.8 – Comparaison entre Padé et splines ($m = 20$).

ρ $[\rho/r_{\max}]$	Padé	$N=4$	8	16	1024
0 [0]	10^{-18}	10^{-17}	10^{-17}	10^{-17}	10^{-17}
10^{-3} [0.001]	2.10^{-8}	7.10^{-7}	2.10^{-7}	2.10^{-7}	2.10^{-7}
10^{-2} [0.011]	9.10^{-5}	1.10^{-5}	6.10^{-7}	2.10^{-7}	4.10^{-7}
10^{-1} [0.111]	1.10^{-3}	2.10^{-3}	2.10^{-3}	3.10^{-7}	1.10^{-8}

TABLE 8.9 – Comparaison entre Padé et Hermite ($m = 60$).

ρ $[\rho/r_{\max}]$	Padé	$N=4$	8	16	1024
0 [0]	10^{-18}	10^{-17}	10^{-17}	10^{-17}	10^{-16}
10^{-3} [0.001]	2.10^{-8}	9.10^{-7}	2.10^{-7}	2.10^{-7}	2.10^{-7}
10^{-2} [0.011]	9.10^{-5}	1.10^{-5}	6.10^{-7}	1.10^{-7}	5.10^{-7}
10^{-1} [0.111]	1.10^{-3}	2.10^{-3}	2.10^{-3}	3.10^{-7}	2.10^{-8}

TABLE 8.10 – Comparaison entre Padé et splines ($m = 60$).

ρ $[\rho/r_{\max}]$	Hermite(4)	(6)	(10)	(18)
0 [0]	10^{-17}	10^{-17}	10^{-17}	10^{-17}
10^{-3} [0.001]	3.10^{-6}	3.10^{-6}	3.10^{-6}	3.10^{-6}
10^{-2} [0.011]	5.10^{-7}	6.10^{-7}	6.10^{-7}	6.10^{-7}
10^{-1} [0.111]	5.10^{-8}	8.10^{-8}	1.10^{-7}	1.10^{-7}

TABLE 8.11 – Interpolation d'Hermite avec différents ordres d'interpolation. Paramètres : $N = 1024$, $m = 1$.

résultats, même avec un nombre relativement faible de points sur le cercle (par exemple avec $\rho = 0.1$ et $N \geq 8$).

5. *La valeur $N = 1024$ est choisie pour avoir une valeur convergée de la gyromoyenne. Nous voyons que très souvent $N = 16$ points sont déjà suffisants pour obtenir une très bonne approximation de cette valeur convergée.*
6. *Dans les Tables 8.3 et 8.4, l'erreur est parfois plus grande pour $N = 1024$ que pour $N = 16$. Ceci pourrait être expliqué par le fait que les 16 points sont plus proches des points du maillage et que l'erreur d'interpolation en (r, θ) est plus grande que l'erreur d'interpolation sur le cercle d'intégration.*
7. *Dans le cas des hauts modes, il nous faut plus de points sur le cercle afin d'évaluer la gyromoyenne lorsque le rayon est grand (Tables 8.9 et 8.10).*
8. *Dans le cas de l'interpolation d'Hermite, nous devons calculer les dérivées à chaque interface de cellules. Ces dérivées sont reconstruites par différences finies centrées d'ordre pair arbitraire. Dans la Table 8.11, ces ordres sont 4, 6, 10 ou 18. Lorsque cet ordre n'est pas spécifié (Tables 8.3, 8.5, 8.7 et 8.9), la valeur par défaut est 4. La Table 8.11 montre que l'on n'obtient pas de meilleurs résultats en augmentant cet ordre de reconstruction.*

8.5 Application aux simulations gyrocinétiques

Les nouvelles méthodes de calcul de la gyromoyenne présentées précédemment ont été testées avec deux codes : (i) la plateforme SELALIB [67] pour le cas simplifié 4D et (ii) le code GYSELA [90] pour le cas 5D classique de référence cyclone DIII-D. Ces deux codes sont basés sur un schéma semi-Lagrangien classique (BSL) avec interpolation par splines cubiques et méthode prédicteur-correcteur.

Dans la suite, les solutions numériques sont calculées en utilisant les équations normalisées. La température est normalisée à T_{e0}, où T_{e0} est définie par le profil de température initial telle que $T_e(r_p)/T_{e0} = 1$. Le temps est normalisé à l'inverse de la fréquence cyclotronique des ions $\omega_c = e_i B_0/m_i$. Les vitesses, y compris la vitesse parallèle, sont exprimées en unités de la vitesse des ions $v_{T0} = \sqrt{T_{e0}/m_i}$, le potentiel électrique est normalisé à T_{e0}/e_i et le champ magnétique est normalisé à B_0. En conséquence, les longueurs sont normalisées au rayon de Larmor $\rho = m_i v_{T0}/e_i B_0$ et le moment magnétique μ à T_{e0}/B_0.

8.5.1 Cas 4D SLAB simplifié

Dans le modèle SLAB 4D, l'équation satisfaite par la fonction de distribution des ions $f(t, r, \theta, z, v)$ suivant le mouvement du centre guide se lit :

$$\partial_t f - \left(\frac{\partial_\theta \mathcal{J}_{\sqrt{2\mu}}\Phi}{r}\right)\partial_r f + \left(\frac{\partial_r \mathcal{J}_{\sqrt{2\mu}}\Phi}{r}\right)\partial_\theta f + v\partial_z f - \left(\partial_z \mathcal{J}_{\sqrt{2\mu}}\Phi\right)\partial_v f = 0. \tag{8.5.1}$$

pour $(r, \theta, z, v) \in [r_{\min}, r_{\max}] \times [0, 2\pi] \times [0, L] \times [-v_{\max}, v_{\max}]$.

Nous nous concentrons sur le transport turbulent entraîné par l'instabilité ITG sans collisions, donc les électrons sont supposés adiabatiques. Dans cette limite, le potentiel électrostatique gyromoyenné $\mathcal{J}_{\sqrt{2\mu}}\Phi$ est la solution de l'équation de quasi-neutralité 3D auto-consistante couplée :

$$\begin{aligned} -\left(\partial_r^2\Phi + \left(\frac{1}{r} + \frac{\partial_r n_0(r)}{n_0(r)}\right)\partial_r\Phi + \frac{1}{r^2}\partial_\theta^2\Phi\right) + \\ \frac{1}{T_e(r)}(\Phi - \lambda\langle\Phi\rangle) = \frac{1}{n_0(r)}\mathcal{J}_{\sqrt{2\mu}}\left(\int f - f_{eq}dv\right), \\ \langle\Phi\rangle = \frac{1}{L}\int_0^L \Phi(r,\theta,z)dz. \end{aligned} \tag{8.5.2}$$

λ est un entier valant 0 ou 1 suivant que l'on converse le terme $\langle\Phi\rangle$ ou non.

Pour traiter ce système d'équations, nous avons utilisé la plate-forme SELALIB [67] avec une méthode semi-Lagrangienne classique avec interpolation par splines cubiques, ainsi qu'une méthode prédicteur-correcteur et l'algorithme de Verlet pour les caractéristiques (voir [86, 77] pour plus de détails). La plate-forme a été améliorée par l'ajout d'un μ fixé et en implémentant les trois opérateurs de gyromoyenne décrits dans les sections 8.2 et 8.3. Dans notre cas, la parallélisation MPI est basée sur des transpositions entre la décomposition de domaine en (r,θ,v) et la décomposition de domaine en z. Dans cette section, les taux d'instabilité numériques sont comparés à ceux déduits de la relation de dispersion obtenue en linéarisant le système d'équations auto-consistant (9.4.2)-(8.5.2).

La dérivation de la relation de dispersion est donnée en annexe G. Nous adaptons un code disponible dans SELALIB, qui calcule les zéros de la relation de dispersion (G.3), comme dans [86], en ajoutant le terme de gyromoyenne. La Fig. 8.10 présente les taux d'instabilité en fonction de μ. Nous obtenons les deux premières courbes en résolvant la relation de dispersion avec $J_0^2(\sqrt{2\mu})$ (courbe en rouge) ou en remplaçant $J_0(\sqrt{2\mu})$ par son approximation de Padé (courbe en vert). Nous avons choisi $\kappa = 1$ dans (G.3). Les deux courbes restantes sont obtenues numériquement avec la méthode de Padé pour l'opérateur de gyromoyenne (courbe en bleu) ou la méthode avec l'interpolation d'Hermite (courbe en magenta). Il en ressort que la pente diminue plus rapidement avec la méthode par interpolation d'Hermite plutôt qu'avec la méthode de Padé. Les pentes obtenues avec le Padé numérique sont différentes de celles obtenues avec la relation de dispersion et l'approximation de Padé puisque les fonctions que nous considérons ici ne sont pas des fonctions de Fourier-Bessel.

Dans les simulations, nous prenons $\lambda = 0$ (cas sans "zonal flow"). La fonction de distribution initiale se lit :

$$\begin{aligned} f(0,r,\theta,z,v) = f_{eq}(r,v) \times \\ \left(1 + \varepsilon\exp\left(-\frac{(r-r_p)^2}{\delta r}\right)\cos\left(\frac{2\pi n}{L}z + m\theta\right)\right) \end{aligned}$$

où la fonction d'équilibre f_{eq} vaut

$$f_{eq}(r,v) = \frac{n_0(r)\exp\left(-\frac{v^2}{2T_i(r)}\right)}{(2\pi T_i(r))^{1/2}}.$$

Les profils T_i, T_e et n_0 sont donnés par :

$$\mathcal{P}(r) = C_{\mathcal{P}} \exp\left(-\kappa_{\mathcal{P}} \delta r_{\mathcal{P}} \tanh\left(\frac{r - r_p}{\delta r_{\mathcal{P}}}\right)\right)$$

où $\mathcal{P} \in \{T_i, T_e, n_0\}$, $C_{T_i} = C_{T_e} = 1$ et

$$C_{n_0} = \frac{r_{\max} - r_{\min}}{\int_{r_{\max}}^{r_{\min}} \exp\left(-\kappa_{n_0} \delta r_{n_0} \tanh\left(\frac{r - r_p}{\delta r_{n_0}}\right)\right) dr}.$$

Nous considérons les paramètres de [86] [Medium case] :

$$\begin{aligned} r_{\min} = 0.1, r_{\max} = 14.5, v_{\max} = 7.32, \kappa_{n_0} = 0.055, \\ \kappa_{T_i} = \kappa_{T_e} = 0.27586, \delta r_{T_i} = \delta r_{T_i} = \frac{\delta r_{n_0}}{2} = 1.45, \\ \varepsilon = 10^{-6}, n = 1, m = 5, \\ L = 1506.759067, r_p = \frac{r_{\min} + r_{\max}}{2}, \delta_r = \frac{4\delta r_{n_0}}{\delta r_{T_i}}. \end{aligned}$$

Les résultats numériques sont donnés dans la Fig. 8.6 – 8.11. Nous considérons ici $N = 1024$ pour Hermite, avec précalcul. Sur la Fig. 4, nous traçons une coupe poloïdale $f(r, \theta, 0, 0)$ ($\mu = 0.5, 1$) au temps 7000 pour Padé et Hermite. Nous observons de plus petites structures pour Padé, ce qui indique que l'instabilité se produit plus rapidement. Des observations similaires peuvent être faites pour la Fig. 5 dans laquelle la même coupe poloidale est tracée, mais pour $\mu = 0, 0.1$. Clairement, quand $\mu = 0$, les deux méthodes donnent les mêmes résultats et pour $\mu = 0.1$, les deux méthodes donnent des résultats très comparables, ce qui est en accord avec les calculs de la relation de dispersion. Enfin, les Fig. 8.8 et 8.9 présentent (pour un maillage moins raffiné 64^4 et $\Delta t = 5$) la coupe poloïdale pour différentes valeurs de μ ($\mu = 0.1, ..., 0.8$). Les différences entre Padé et Hermite deviennent de plus en plus significatives quand μ augmente (voir en particulier le cas $\mu = 0.8$). La Fig. 8.10 montre l'évolution temporelle de $\int_{r_{\min}}^{r_{\max}} \int_0^{2\pi} \Phi(r, \theta, 0) r dr d\theta$ dans la phase linéaire. La gyromoyenne tend à réduire le taux d'instabilité ; plus μ est grand, plus ce taux est faible. La comparaison avec la solution de la relation de dispersion confirme que le taux d'instabilité est plus faible dans le cas de l'interpolation d'Hermite. Nous présentons l'évolution temporelle de $\int_{r_{\min}}^{r_{\max}} \int_0^{2\pi} \Phi(r, \theta, 0) r dr d\theta$ dans la phase non linéaire (fig. 8.11). Dans cette phase, l'instabilité se développe plus rapidement dans le cas Padé que dans le cas de l'interpolation d'Hermite. En outre, lorsque μ est petit, on peut voir que l'instabilité apparait de manière plus lente.
Les résultats de temps (Tables 8.12 and 8.13) montrent que le choix de l'opérateur de gyromoyenne n'est pas très influent dans le temps total. En effet, le calcul de la gyromoyenne est un problème $3D$ dans un environnement $4D$.

8.5.2 Benchmark avec le classique cas test 5D Cyclone DIII-D

Dans cette partie, l'opérateur de gyromoyenne basé sur les interpolations par splines cubiques et par Hermite a été mis en œuvre dans le code GYSELA [90] et comparé à l'approximation de Padé existante. Le problème Vlasov-Poisson 5D considéré est celui décrit par

μ	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Herm.	15	16	15	16	16	16	16	16	15
Padé	15	15	14	15	17	15	16	16	15

TABLE 8.12 – Temps (en min.) sur HPC, mésocentre de l'Université de Strasbourg, avec 64 processeurs (8 noeuds). $N_r \times N_\theta \times N_z \times N_v = 64 \times 64 \times 32 \times 64$, $\Delta t = 5$, 1600 itérations. Pour Hermite, avec précalcul : $N = 1024$.

μ	Hermite	Padé
0.5	12524	13326
1	12556	13454

TABLE 8.13 – Temps final (en s.) atteint pour une simulation de 24 heures avec $N_r \times N_\theta \times N_z \times N_v = 128 \times 256 \times 128 \times 128$, $\Delta t = 2$. Sur le centre de simulation Helios, centre international de recherche sur l'énergie par fusion. Supercalculateur avec 128 processeurs (8 noeuds ; chaque noeud a 16 threads). Pour Hermite, avec précalcul : $N = 1024$.

les équations (6.5.1)-(6.5.5) et (6.5.7). Dans la suite, tous les résultats numériques présentés sont exprimés en unités normalisées du code sauf dans la figure 8.5. Dans le code GYSELA, la température est normalisée à T_{e0}, où T_{e0} est définie par le profil initial de température telle que $T_e(r_p)/T_{e0} = 1$. Le temps est normalisé à l'inverse de la fréquence cyclotron ionique $\omega_c = e_i B_0/m_i$. Les vitesses, y compris la vitesse parallèle, sont exprimées en unités de la vitesse d'ion $v_{T0} = \sqrt{T_{e0}/m_i}$, le potentiel électrique est normalisée à T_{e0}/e_i et le champ magnétique est normalisé à B_0. En conséquence, les longueurs sont normalisées au rayon de Larmor $\rho_s = m_i v_{T0}/e_i B_0$ et le moment magnétique μ à T_{e0}/B_0.

Tout d'abord, nous comparons les deux méthodes d'un point de vue numérique. Les comparaisons numériques ont été réalisées sur un benchmark linéaire basé sur le cas classique cyclone DIII-D [88]. Ce benchmark typique avait déjà été utilisé il y a plusieurs années pour valider le code GYSELA [90]. Pour les tests actuels, les mêmes paramètres que dans la section 4 de [90] ont été utilisés excepté pour la taille de la simulation. Nous considérons le maillage suivant :

$$N_r = 256,\ N_\theta = 256,\ N_\varphi = 64,\ N_{v_\parallel} = 64,\ N_\mu = 8 \tag{8.5.3}$$

Pour les méthodes d'Hermite et de splines cubiques, $N = 32$ points de quadrature sont utilisés. Cinq simulations ont été effectuées pour les trois opérateurs de gyromoyenne différents (Padé, Hermite et splines cubiques). Chaque simulation correspond à l'excitation d'un mode initialement instable différent (m, n) avec m le mode poloïdal et n le mode toroïdal. Les résultats obtenus avec l'interpolation par splines cubiques ne sont pas détaillés car ils sont très similaires aux résultats obtenus avec l'interpolation d'Hermite. Les valeurs numériques des taux d'accroissements linéaires normalisées associées à chaque mode (m, n) sont données dans la Table 8.14. Dans la Figure 8.5, les résultats sont représentés sous la forme proposée par Dimits (cf. figure 1 dans [88]).

Dans ces simulations, les valeurs minimales et maximales de μ sont respectivement $\mu_{\min} = 0.143$ et $\mu_{\max} = 7$. Puis, en utilisant la relation $k_\theta\sqrt{2\mu} \sim m/r_p\sqrt{2\mu} = 0.5a\sqrt{2\mu}$ où a est la

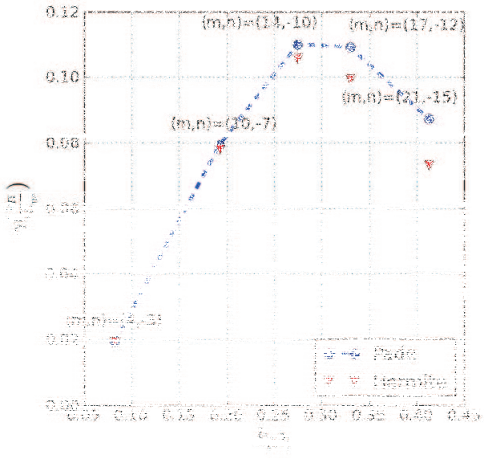


FIGURE 8.5 – Taux d'instabilité linéaire en fonction de $k_\theta \rho_i$ pour le cas CYCLONE DIII-D avec : (cercles) l'approximation de Padé et (triangles) la gyromoyenne en utilisant l'interpolation d'Hermite.

(m,n)	$(4,-3)$	$(10,-7)$	$(14,-10)$
$k_\theta\sqrt{(2\mu_{\max})}$	0.98	1.98	2.77
γ with Padé	$1.3e^{-4}$	$5.7e^{-4}$	$7.9e^{-4}$
γ with Herm.	$1.3e^{-4}$	$5.6e^{-4}$	$7.6e^{-4}$

(m,n)	$(17,-12)$	$(21,-15)$
$k_\theta\sqrt{(2\mu_{\max})}$	3.36	4.15
γ with Padé	$7.8e^{-4}$	$6.2e^{-4}$
γ with Herm.	$7.16e^{-4}$	$5.2e^{-4}$

TABLE 8.14 – Taux d'accroissement linéaires normalisés à Ω_{C0} pour le cas de base Cyclone DIII-D.

dimension radiale de la boîte de simulation qui est égale à 100, l'intervalle des valeurs du rayon de Larmor pour l'opérateur de gyromoyenne si situe entre $0.028\, m\sqrt{\mu_{\min}}$ et $0.028\, m\sqrt{\mu_{\max}}$. La valeur maximale pour chaque mode (m,n) est donnée dans la Table 8.14. Nous observons que les résultats sont similaires entre Padé et Hermite pour les modes $(m,n) = (4,-3)$ et $(m,n) = (10,-7)$ et que les taux d'accroissement linéaires deviennent plus petits avec l'opérateur de gyromoyenne d'Hermite pour m plus grand que 14. Ceci est en accord avec ce qui a été observé dans la section précédente et ce qui est montré dans la figure 8.1. Le développement de Padé approche correctement la fonction de Bessel $J_0(k_\theta\rho)$ pour $k_\theta\rho < 1$, mais la surestime pour $1 < k_\theta\rho < 5$. Des simulations non linéaires en temps long doivent être exécutées à l'avenir pour analyser l'impact sur les grands nombres d'onde de Fourier $(k_\theta\rho > 5)$.

Optimisation parallèle pour l'opérateur de gyromoyenne basé sur Hermite

Un autre challenge pour Hermite ou pour les splines cubiques était d'être compétitif en termes de temps de calcul avec l'opérateur de Padé optimisé utilisé jusqu'à présent pour les simulations de GYSELA. Une analyse des performances est présentée dans ce qui suit.

Pour mesurer ces performances, différents cas test pour 4 itérations ont été lancés sur la machine HELIOS[1]. Les noeuds de calcul utilisés durant nos tests sont équipés de deux processeurs Intel Xeon E5-2450 2.10GHz avec 16 cœurs par noeud. Dans la Table 8.15, les différents cas test ont été réalisés sur les trois maillages (8.5.4), (8.5.5) et (8.5.6) :

$$N_r = 128,\ N_\theta = 128,\ N_\varphi = 32,\ N_{v_\parallel} = 16,\ N_\mu = 4 \tag{8.5.4}$$

$$N_r = 256,\ N_\theta = 256,\ N_\varphi = 32,\ N_{v_\parallel} = 16,\ N_\mu = 4 \tag{8.5.5}$$

$$N_r = 512,\ N_\theta = 512,\ N_\varphi = 32,\ N_{v_\parallel} = 16,\ N_\mu = 4. \tag{8.5.6}$$

Nous pouvons remarquer qu'entre les trois maillages, la seule différence est le nombre de points dans les directions radiale et poloïdale. Le maillage (8.5.5) a quatre fois plus de points que le maillage (8.5.4), et de même entre les maillages (8.5.6) et (8.5.5). Nous avons choisi d'augmenter le nombre de points dans ces directions parce que la taille du plan poloïdal a le plus d'impact sur le temps d'exécution de l'opérateur de gyromoyenne. Pour éviter l'effet du réseau sur ces mesures de performances, les paramètres des différentes exécutions ont été ajustés pour s'adapter à un seul nœud de calcul de la machine HELIOS. Pour simplifier l'interprétation des résultats, les exécutions ont été effectuées avec un seul thread.

Sur la Table 8.15, nous pouvons voir l'évaluation de 3 versions de l'opérateur de gyromoyenne : Padé, Hermite (sans et avec précalcul) et les splines cubiques (avec et sans précalcul). En outre, le code GYSELA peut être exécuté sans gyromoyenne. Le temps d'exécution dans cette configuration nous permet alors d'en déduire avec précision le pourcentage de temps que nécessite l'opérateur de gyromoyenne.

Comme prévu, pour toutes les méthodes, plus le nombre de points dans le plan poloïdal est élevé, plus la proportion du temps d'exécution de la gyromoyenne est conséquente par rapport à la durée totale de la simulation. Sur tous les maillages, nous pouvons constater d'importantes accélérations pour les versions avec précalcul comparées aux versions sans précalcul dans le cas des méthodes d'Hermite et des splines. Malgré ces accélerations, les méthodes basées sur l'interpolation sont en moyenne 9 fois plus lentes que l'approximation de Padé.

Selon la Table 8.15, avec un thread, en fonction de la taille du maillage, la gyromoyenne basée sur l'interpolation peut s'exécuter aussi vite qu'avec Padé ce qui est un bon argument de son intégration en vue de la mise en production dans GYSELA.

Pour le dernier aspect de performance de notre étude, nous nous concentrons sur l'impact du nombre de threads. En fait, le code GYSELA est parallélisé en MPI et OPENMP. Le but de cette parallélisation hybride consiste à utiliser efficacement les supers-ordinateurs actuels. Dans ce contexte, la version de calcul de gyromoyenne par Hermite a été améliorée pour pouvoir être appelée simultanément à partir de plusieurs threads (thread-safe). L'effort de

1. http ://www.top500.org/system/177449

développement s'est concentré uniquement sur le cas d'Hermite parce qu'il comporte au moins deux aspects attractifs par rapport à l'interpolation par splines cubiques : *(i)* la mise en œuvre de l'interpolation d'Hermite nous permet de choisir le degré de reconstruction des dérivées en nous donnant donc la possibilité d'améliorer la qualité des résultats et *(ii)* la version Hermite nécessite seulement quelques points (selon le degré de reconstruction des dérivées) autour de la position cible de l'interpolation tandis que l'approche par splines cubiques est non locale.

Pour évaluer le comportement de la gyromoyenne, nous avons fait une série d'exécutions de 4 itérations avec un nombre différent de threads sur le maillage (8.5.6). Nous avons choisi un faible nombre d'itérations afin d'avoir des temps d'exécution raisonnables. Les différentes exécutions ont été encore effectuées sur la machine HELIOS avec 1, 2, 4 et 8 threads et respectivement 16, 32, 64 et 128 cœurs.

La Table 8.16 contient différentes mesures de performance pour les trois opérateurs de gyromoyenne : *(i)* Padé, *(ii)* Hermite avec précalculs et *(iii)* sans gyromoyenne. Comme précédemment, le temps d'exécution et la proportion de temps de chaque méthode de gyromoyenne sont donnés. En outre, l'efficacité relative est donnée et calculée par :

$$\left(1 + \frac{t_{ref} - \#th_{targ}.\, t_{targ}}{t_{ref}}\right) \times 100 \tag{8.5.7}$$

où t_{ref} est le temps de l'exécution de référence, pour nous l'exécution sur un thread, t_{targ} et $\#th_{targ}$ respectivement le temps d'exécution et le nombre de threads de l'exécution. Plus l'efficacité relative est grande, plus la qualité de la parallélisation est importante.

Tout d'abord, nous pouvons noter que le temps total d'exécution pour les trois configurations scale avec le nombre de threads. Ensuite, les méthodes d'Hermite et de Padé ont un comportement similaire en considérant l'efficacité relative lorsque le nombre de threads augmente. Dans le cas final avec 8 threads, les deux méthodes gardent une efficacité relative au-dessus de 80%. Finalement, en tenant compte de la proportion du temps d'exécution d'une méthode pour chaque nombre de threads, nous remarquons qu'elle reste relativement constante. Comme dans le cas de l'analyse précédente, la gyromoyenne basée sur l'interpolation d'Hermite est en moyenne 9 fois plus lente que l'approximation de Padé.

L'analyse de performances précédente ne tient pas compte du temps d'exécution des diagnostics de GYSELA. L'opérateur de gyromoyenne a un grand impact sur leur temps d'exécution, mais la mesure du temps d'exécution de cette partie du code est difficile à cause de l'écriture des fichiers de sortie sur le disque qui peuvent introduire un comportement aléatoire. En effet, certains diagnostics appliquent la gyromoyenne sur toute la fonction de distribution des centres-guide pour obtenir des informations sur la distribution des particules. Il y a un champ d'amélioration possible pour l'opérateur de gyromoyenne basé sur l'interpolation afin de mieux traiter ce genre d'opérations. Pour le futur, nous souhaitons optimiser la méthode d'Hermite afin de raccourcir son temps de calcul et ainsi être plus compétitif par rapport à la méthode de Padé.

8.6 Conclusion

Nous avons validé le calcul de la gyromoyenne en géométrie polaire. Des comparaisons sont faites avec l'approximation classique de Padé, en considérant d'une part des cas test analytiques, dont on connaît la solution exacte, et d'autre part des simulations gyrocinétiques de

base : un modèle drift-kinetic $4D$ avec un rayon de Larmor et le benchmark linéaire classique DIII-D. Nous constatons que, dans le cas linéaire, les différences avec Padé sont importantes en prenant une géométrie SLAB et un rayon relativement grand. En outre, l'introduction de l'opération de gyromoyenne tend à diminuer le taux d'instabilité et cela est amplifié en considérant l'opérateur de gyromoyenne direct, au lieu de l'approximation de Padé. L'analyse linéaire prédit un comportement similaire lorsque nous comparons l'approximation de Padé et la fonction de Bessel J_0 pour $k\rho < 1$, mais une diminution du taux d'accroissement quand $1 < k\rho < 5$. Notons que le résultat reste au niveau qualitatif, comme ici, en géométrie polaire, la multiplication par la fonction de Bessel J_0 n'est pas la solution exacte (excepté pour les fonctions de Fourier-Bessel), et elle diffère de la géométrie cartésienne . Des simulations non-linéaires en temps long devraient être réalisées dans le futur afin de voir l'impact sur de grands nombres d'ondes.

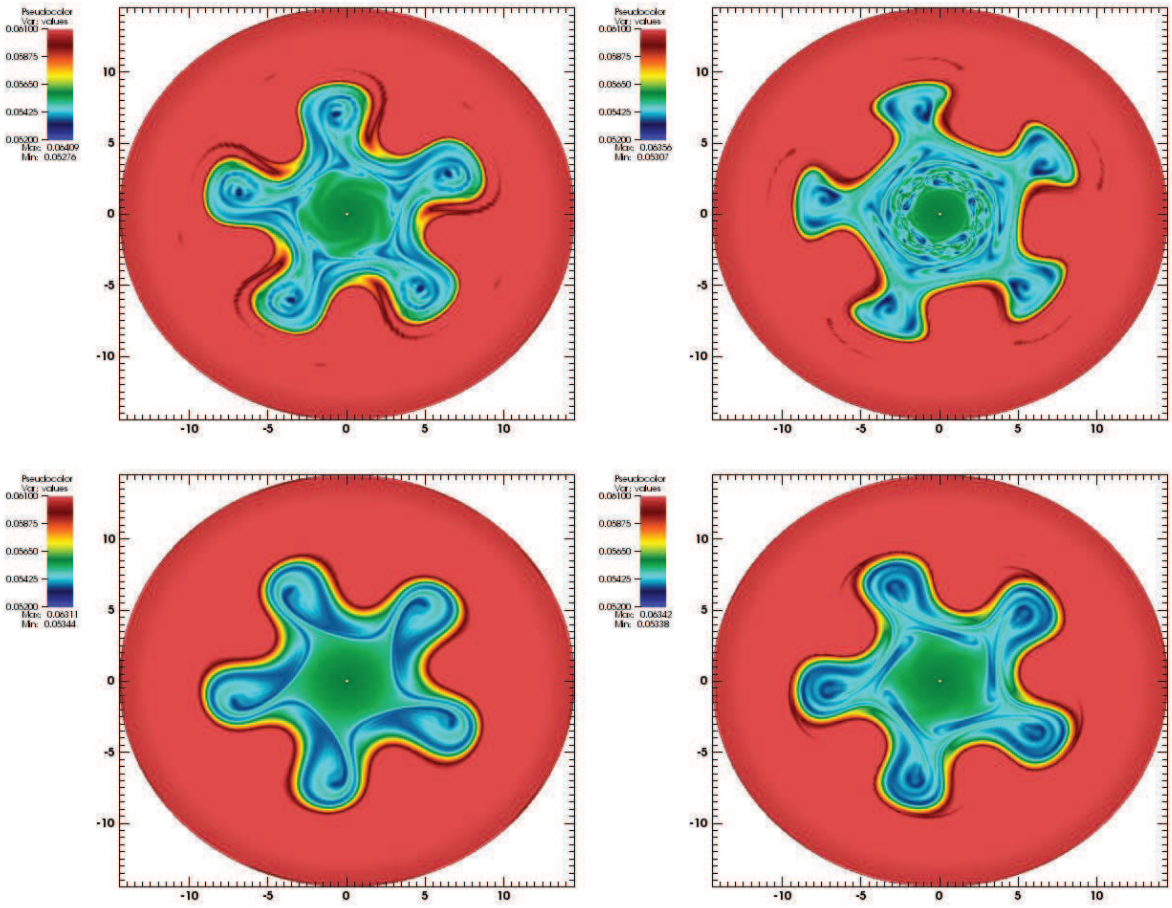


FIGURE 8.6 – Coupe poloïdale $f(r, \theta, z = 0, v_{\|} = 0)$ au temps $T = 7000$ pour $128 \times 256 \times 128 \times 128, \Delta t = 2$. En haut (de gauche à droite) : $\mu = 0.5$ avec Hermite et puis Padé ; en bas (de gauche à droite) : $\mu = 1$, avec Hermite et puis Padé.

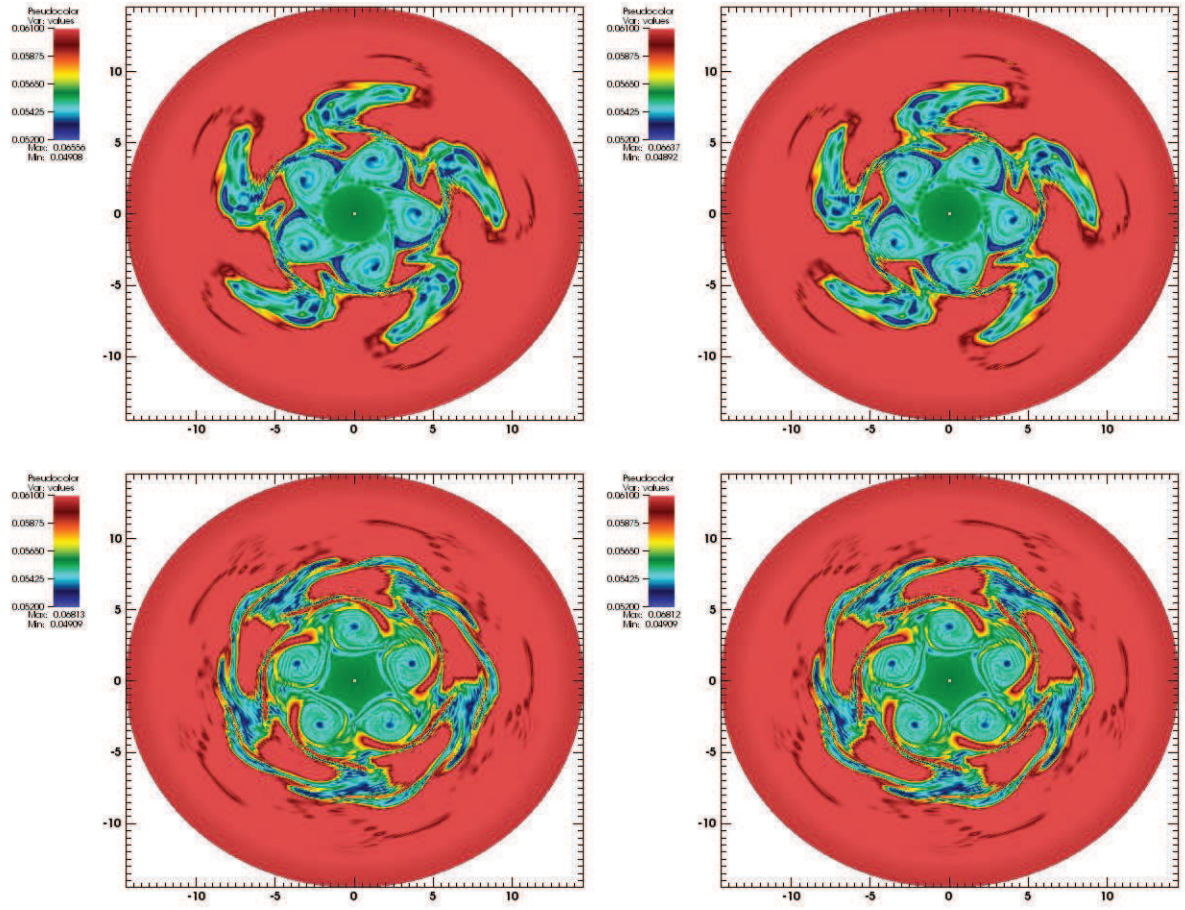


FIGURE 8.7 – Coupe poloïdale $f(r, \theta, z = 0, v_{\|} = 0)$ au temps $T = 5000$ pour $128 \times 128 \times 128 \times 128, \Delta t = 1$. En haut (de gauche à droite) : $\mu = 0.1$ avec Hermite et puis Padé ; en bas (de gauche à droite) : $\mu = 0$, avec Hermite et puis Padé.

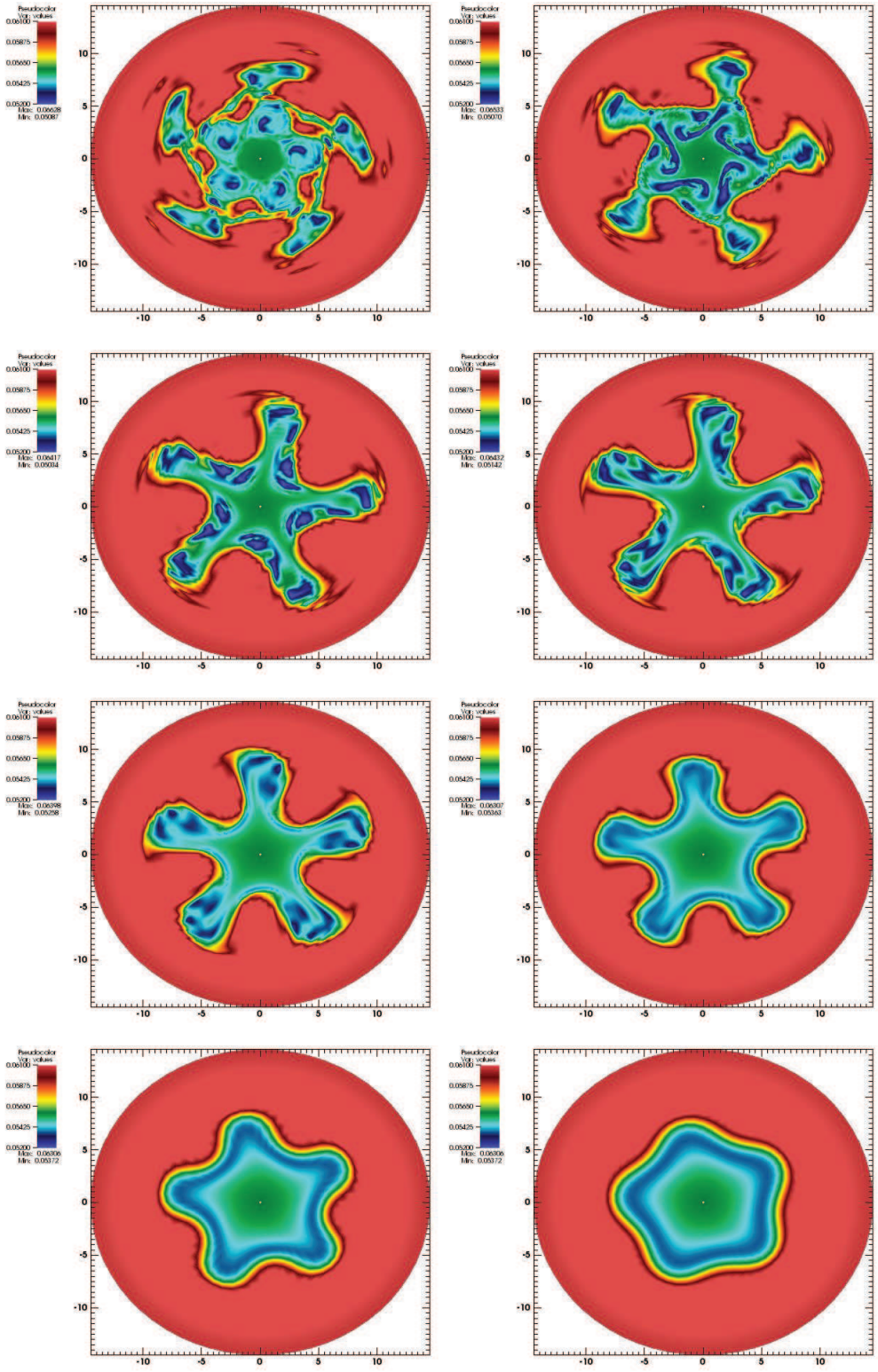


FIGURE 8.8 – coupe poloïdale $f(r, \theta, 0, 0)$ au temps $T = 5000$ avec $64 \times 64 \times 32 \times 64$ points et $\Delta t = 5$, Hermite avec $\mu = 0.1, \ldots, 0.8$ (de gauche à droite et de haut en bas).

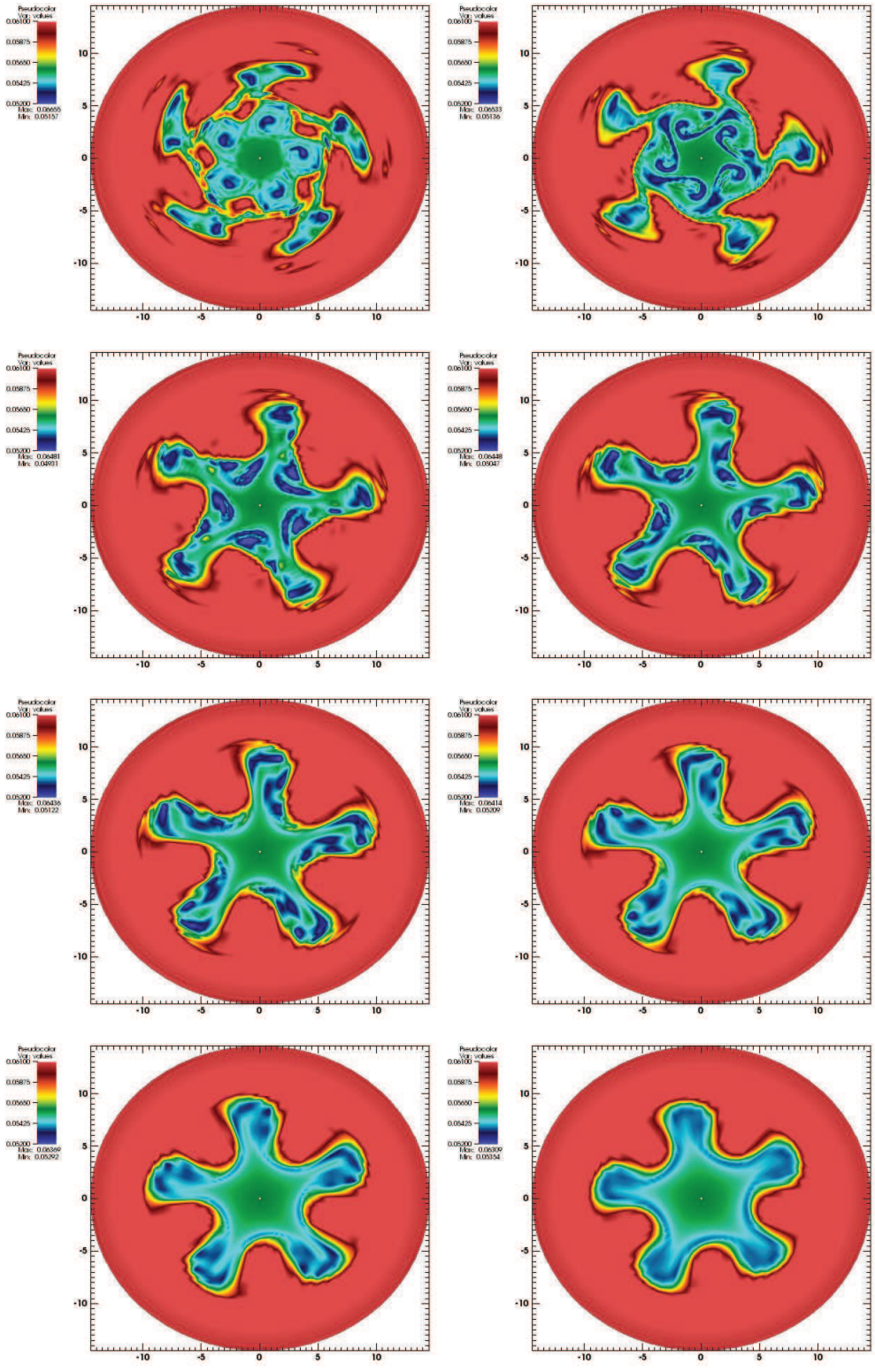


FIGURE 8.9 – coupe poloïdale $f(r, \theta, 0, 0)$ au temps $T = 5000$ avec $64 \times 64 \times 32 \times 64$ points et $\Delta t = 5$, Padé avec $\mu = 0.1, \ldots, 0.8$ (de gauche à droite et de haut en bas).

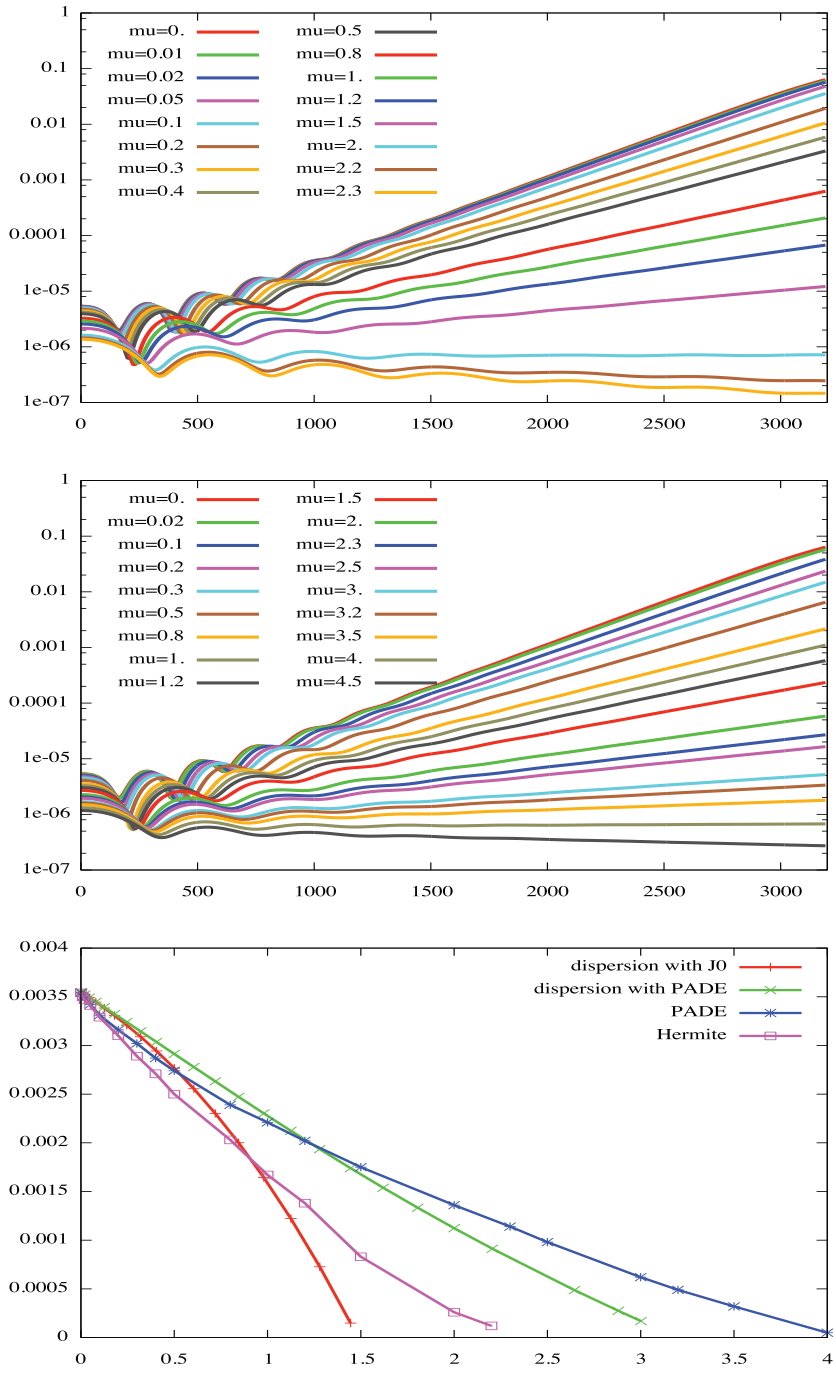


FIGURE 8.10 – Evolution en temps de $\int_{r_{\min}}^{r_{\max}} \int_0^{2\pi} \Phi(r,\theta,0) r dr d\theta$: Hermite (en haut), Padé (milieu). En bas : taux d'instabilité en fonction de μ ; comparaison entre la solution de la relation de dispersion (G.3) (en utilisant $J_0(\sqrt{2\mu})$ ou son approximation de Padé) et les résultats numériques.

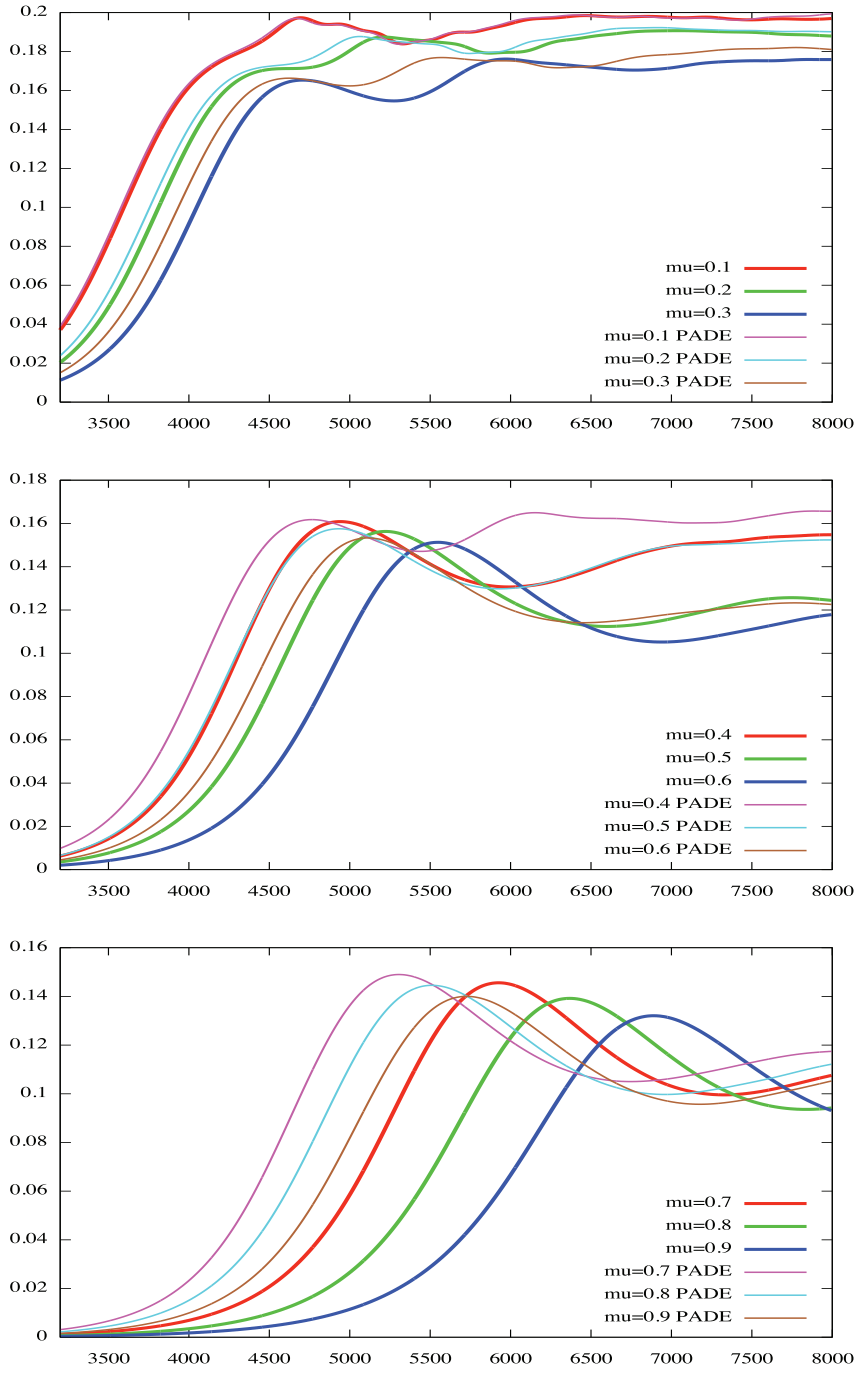


FIGURE 8.11 – Evolution en temps de $\int_{r_{\min}}^{r_{\max}} \int_0^{2\pi} \Phi(r,\theta,0) r dr d\theta$. Comparaison Hermite/Padé.

		Méthode de gyromoyenne					
		Padé	Hermite		Spline		`Désactivé`
			Sans	Avec	Sans	Avec	
Mail. (8.5.4) :	Temps tot. d'exéc. (sec.)	26.71	46.24	29.00	47.70	28.72	`26.33`
	% exéc. de la gyro.	1.4 %	75.6 %	10.2 %	81.2 %	9.1 %	$\emptyset$
Mail. (8.5.5) :	Temps tot. d'exéc. (sec.)	105.98	186.83	120.55	190.25	127.12	`104.78`
	% exéc. de la gyro.	1.1 %	78.3 %	15.0 %	81.6 %	21.3 %	$\emptyset$
Mail. (8.5.6) :	Temps tot. d'exéc. (sec.)	453.52	760.85	525.18	793.47	594.18	`444.20`
	% exéc. de la gyro.	2.1 %	71.3 %	18.2 %	78.6 %	33.8 %	$\emptyset$

TABLE 8.15 – Temps d'exécution et proportion des différents opérateurs de gyromoyenne sur trois maillages différents.

		Méthode de gyromoyenne		
		Padé	Hermite	`Désactivé`
#thread : 1	Temps tot. d'exéc. (sec.)	454.74	526.75	443.85
	Efficacité relative	100.0 %	100.0 %	∅
	% exéc. de la gyro.	2.5 %	18.7 %	∅
#thread : 2	Temps tot. d'exéc. (sec.)	232.53	265.67	227.86
	Efficacité relative	97.8 %	99.1 %	∅
	% exéc. de la gyro.	2.1 %	16.6 %	∅
#thread : 4	Temps tot. d'exéc. (sec.)	121.07	141.83	117.83
	Efficacité relative	93.9 %	92.8 %	∅
	% exéc. de la gyro.	2.7 %	20.4 %	∅
#thread : 8	Temps tot. d'exéc. (sec.)	66.37	77.66	64.98
	Efficacité relative	85.6 %	84.8 %	∅
	% exéc. de la gyro.	2.1 %	19.5 %	∅

TABLE 8.16 – Sur le maillage (8.5.6) : temps d'exécution en secondes, efficacité relative en pourcentage et proportion des différentes versions de l'opérateur de gyromoyenne en fonction du nombre de threads.

Chapitre 9

Quasi-neutralité

Dans le chapitre précédent, nous avons développé une méthode numérique de calcul de l'opérateur de gyromoyenne :

$$\mathcal{J}_\rho(f)(\mathbf{x}) := \frac{1}{2\pi}\int_0^{2\pi} f(\mathbf{x}+\boldsymbol{\rho})d\alpha, \qquad \boldsymbol{\rho} = \rho(\cos(\alpha),\sin(\alpha))$$

sur un maillage polaire. Nous développons ici un solveur pour l'équation de quasi-neutralité utilisant cette méthode. Nous comparerons ce solveur à la méthode classique par Padé sur des cas tests analytiques. Nous considérons également un modèle simplifié de l'équation de quasi-neutralité avec un μ (au lieu d'une intégrale en μ) sur lequel nous comparerons le nouveau solveur avec le solveur par Padé dans le cadre de simulations gyrocinétiques.

9.1 Dérivation de l'équation de quasi-neutralité

Dans cette partie, nous dérivons un modèle de quasi-neutralité permettant de calculer le potentiel électrique $\Phi(\mathbf{x},t)$ où $\mathbf{x} = (r,\theta,z) \in [0,L_r]\times[0,2\pi]\times[0,L_z]$ sont les coordonnées de la position des particules. Nous désignons la densité des ions par n_i, la densité des électrons par n_e et n_0 correspond à la densité des ions et des électrons à l'équilibre. Nous considérerons que cette densité à l'équilibre varie uniquement de manière radiale : $n_0 = n_0(r)$. Nous notons $T_i = T_i(r)$ la température des ions à l'équilibre et $T_e = T_e(r)$ celle des électrons.

Les électrons suivent une distribution de Boltzmann :

$$n_e(\mathbf{x}) = n_0(r)\exp\left(\frac{-(\Phi(\mathbf{x}) - \langle\Phi\rangle(r,\theta))}{T_e(r)}\right)$$

où le potentiel moyen suivant z noté $\langle\Phi\rangle$ est défini par :

$$\langle\Phi\rangle(r,\theta) := \frac{1}{L_z}\int_0^{L_z}\Phi(r,\theta,z)dz.$$

Puisque le potentiel est petit devant l'énergie cinétique des électrons, nous pouvons linéariser :

$$n_e(\mathbf{x}) = n_0(r)\left(1 - \frac{1}{T_e(r)}(\Phi(\mathbf{x}) - \langle\Phi\rangle(r,\theta))\right).$$

Par ailleurs, la densité des ions n_i peut être écrite au premier ordre gyrocinétique par :

$$n_i(\mathbf{x}) = \int \mathcal{J}_\rho(f+g)(\mathbf{x}, v_z)\rho d\rho dv_z, \qquad \rho = \sqrt{2\mu}$$

où la fonction g correspond à la correction :

$$g(\mathbf{x}, \rho) := \partial_\mu F_M(r, \rho)[\Phi(\mathbf{x}) - \mathcal{J}_\rho(\Phi)(\mathbf{x})], \qquad F_M(r, \rho) := \frac{n_0(r)}{T_i(r)} \exp(-\rho^2/(2T_i)).$$

de telle sorte à ce que

$$\frac{1}{n_0(r)} \int_0^{+\infty} F_M(r, \rho)\rho d\rho = 1.$$

En notant par $\bar{n}_i$ la densité relative à la fonction de distribution des ions gyromoyennée :

$$\bar{n}_i(\mathbf{x}) := \int \mathcal{J}_\rho(f)(\mathbf{x}, v_z)\rho d\rho dv_z,$$

la densité des ions n_i peut alors se réécrire :

$$n_i(\mathbf{x}) = \int \mathcal{J}_\rho(f)(\mathbf{x}, v_z)\rho d\rho dv_z + \int \mathcal{J}_\rho(g)(\mathbf{x})\rho d\rho.$$

Concernant le terme de correction, nous avons :

$$g(\mathbf{x}, \rho) \;=\; -\frac{1}{T_i(r)} F_M(r, \rho)[\Phi(\mathbf{x}) - \mathcal{J}_\rho(\Phi)(\mathbf{x})].$$

Puisque les variations spatiales de F_M sont négligeables, nous pouvons supposer que $\mathcal{J}_\rho(F_M) = F_M$ et ainsi :

$$\begin{aligned} \mathcal{J}_\rho(g)(\mathbf{x}, \rho) &= -\frac{1}{T_i(r)} F_M(r, \rho)[\Phi(\mathbf{x}) - \mathcal{J}_\rho^2(\Phi)(\mathbf{x})] \\ \int \mathcal{J}_\rho(g)(\mathbf{x}, \rho)\rho d\rho &= -n_0(r) \left(\Phi(\mathbf{x}) - \frac{1}{T_i(r)} \int \mathcal{J}_\rho^2(\Phi)(\mathbf{x}) \exp(-\rho^2/(2T_i(r)))\rho d\rho \right). \end{aligned}$$

En posant

$$\tilde{\Phi}(\mathbf{x}) := \frac{1}{T_i(r)} \int_{\mathbb{R}^+} \mathcal{J}^2_{\sqrt{2\mu}}(\Phi)(\mathbf{x}) \exp(-\mu/T_i(r)) d\mu,$$

nous obtenons :

$$n_i(\mathbf{x}) = \bar{n}_i(\mathbf{x}) - n_0(r)(\Phi(\mathbf{x}) - \tilde{\Phi}(\mathbf{x})).$$

Sous l'hypothèse de la limite quasi-neutre, la densité des ions est égale à celle des électrons : $n_i = n_e$. Cette hypothèse nous conduit à l'équation de quasi-neutralité :

$$\frac{n_0}{T_i} \int_{\mathbb{R}^+} (\Phi - \mathcal{J}^2_{\sqrt{2\mu}}(\Phi)) \exp(-\mu/T_i) d\mu - \frac{n_0}{T_e}(\Phi - \langle \Phi \rangle) = \bar{n}_i - n_0. \tag{9.1.1}$$

9.2 Solveur par interpolation

Nous présentons une méthode de résolution de l'équation de quasi-neutralité (9.1.1) directement basée sur la méthode de calcul de la gyromoyenne décrite dans le Chapitre 8 qui consistait à interpoler la fonction de distribution en N points uniformément répartis sur le cercle de Larmor. Pour cela, considérons un maillage polaire uniforme sur le domaine $[r_{\min}, r_{\max}] \times [0, 2\pi]$ comprenant $N_r \times N_\theta$ cellules :

$$C_{ij} = [r_i, r_{i+1}] \times [\theta_j, \theta_{j+1}], \qquad i = 0, \ldots, N_r, \ j = 0, \ldots, N_\theta - 1$$

où

$$\begin{aligned} r_i &= r_{\min} + i\frac{r_{\max} - r_{\min}}{N_r}, \qquad i = 0, ..., N_r, \\ \theta_j &= j\frac{2\pi}{N_\theta}, \qquad j = 0, ..., N_\theta. \end{aligned}$$

Le calcul de la gyromoyenne s'écrit, pour (r_i, θ_j) un point du maillage polaire :

$$\mathcal{J}_\rho(\Phi)(r_i, \theta_j) \simeq \frac{1}{N}\sum_{\ell=0}^{N-1} \mathcal{P}(\Phi)\left(r_i\cos(\theta_j) + \rho\cos\left(\frac{2\ell\pi}{N}\right), r_i\sin(\theta_j) + \rho\sin\left(\frac{2\ell\pi}{N}\right)\right),$$

où $\mathcal{P}$ est un opérateur d'interpolation. Dans la suite, nous utiliserons l'interpolation par splines cubiques. Nous effectuons une projection radiale sur le bord du domaine pour les points hors du domaine et nous considérerons des conditions 2π-périodiques en θ.

Nous détaillons ci-dessous les étapes de construction du solveur. Pour cela, notons

$$\begin{aligned} \Phi_{i,j} &:= \Phi(r_i, \theta_j), \qquad i = 0..N_r, \ j = 0..N_\theta - 1 \\ \mathcal{J}_\rho\Phi_{i,j} &:= \mathcal{J}_\rho(\Phi)(r_i, \theta_j), \qquad i = 0..N_r, \ j = 0..N_\theta - 1 \\ \phi &:= {}^t(\Phi_{0,0}, ..., \Phi_{0,N_\theta-1}, \Phi_{1,0}, ..., \Phi_{1,N_\theta-1}, ..., \Phi_{N_r,1}, ..., \Phi_{N_r,N_\theta-1}) \\ \mathcal{J}_\rho(\phi) &:= {}^t(\mathcal{J}_\rho\Phi_{0,0}, ..., \mathcal{J}_\rho\Phi_{0,N_\theta-1}, \mathcal{J}_\rho\Phi_{1,0}, ..., \mathcal{J}_\rho\Phi_{1,N_\theta-1}, ..., \mathcal{J}_\rho\Phi_{N_r,0}, ..., \mathcal{J}_\rho\Phi_{N_r,N_\theta-1}). \end{aligned}$$

1. Construction de la matrice $A^{spl} \in \mathcal{M}_{(N_r+3)\times N_\theta,(N_r+1)\times N_\theta}(\mathbb{R})$ telle que $S = A^{spl}\phi$ soit le vecteur des coefficients de splines. Nous considérons des splines de type Hermite avec condition nulle sur la dérivée aux bords en r et des conditions périodiques en θ. La matrice A^{spl} ne dépend pas du rayon de Larmor.

2. Pour chaque rayon de Larmor $\rho_j = \sqrt{2\mu_j}$, construction de la matrice $A^{contr}_{\rho_j} \in \mathcal{M}_{(N_r+1)\times N_\theta,(N_r+3)\times N_\theta}(\mathbb{R})$ donnant la contribution de la gyromoyenne de rayon ρ_j en chaque point en fonction des coefficients de splines. Nous avons donc $\mathcal{J}_{\rho_j}(\phi) = A^{contr}_{\rho_j} S$ et la matrice de la gyromoyenne pour le rayon de Larmor ρ_j est alors donnée par $G_{\rho_j} = A^{contr}_{\rho_j} A^{spl}$. La matrice de la double gyromoyenne s'obtient par $B_{\rho_j} = G^2_{\rho_j}$. Nous remarquons que pour un maillage de ρ_j donné, ces matrices peuvent être calculées une fois pour toute.

3. Evaluation de l'intégrale par quadrature en μ :

$$\begin{aligned}\int_{\mathbb{R}^+}(\Phi-\mathcal{J}^2_{\sqrt{2\mu}}(\Phi))e^{-\mu/T_i}d\mu &\approx \int_0^{\mu_{\max}}(\Phi-\mathcal{J}^2_{\sqrt{2\mu}}(\Phi))e^{-\mu/T_i}d\mu\\ &\approx \sum_{j=1}^{p}c_j(\Phi-\mathcal{J}^2_{\sqrt{2\mu_j}}(\Phi))e^{-\mu_j/T_i}\\ &\approx \left[\sum_{j=1}^{p}c_j(\mathrm{Id}-B_{\rho_j})e^{-\mu_j/T_i}\right]\Phi.\end{aligned}$$

Nous utilisons les méthodes de quadrature suivantes :
— Méthode des rectangles à gauche : $p\geq 1$, $c_j=\frac{\mu_{\max}}{p}$ et $\mu_j=(j-1)\frac{\mu_{\max}}{p}$.
— Méthode de Gauss-Legendre (formule composite).

4. Inversion de la matrice de l'opérateur de quasi-neutralité par décomposition LU.

Remarque 9.2.1.

1. *Il est à noter que toutes les étapes ci-dessus sont effectuées en précalcul.*
2. *Une méthode d'intégration est également présentée dans [105] basée sur l'approximation de la fonction Γ_0.*

Nous construisons la matrice de l'opérateur de quasi-neutralité dans la base de Fourier vectorielle. En effet, le calcul est plus efficace dans cette base puisque les produits matriciels nécessaires à la construction de la matrice de double gyromoyenne se font entre des matrices diagonales par blocs. Cette méthode a déjà été utilisée par exemple dans le chapitre 5. Plus précisément, la périodicité en θ assure que les matrices $A^{contr}_{\rho_j}$ et A^{spl} sont circulantes par blocs :

$$A^{spl}=\begin{pmatrix}A^{spl}_0 & A^{spl}_1 & \dots & A^{spl}_{N_\theta-1}\\ A^{spl}_{N_\theta-1} & \ddots & \ddots & \vdots\\ \vdots & \ddots & \ddots & A^{spl}_1\\ A^{spl}_1 & \dots & A^{spl}_{N_\theta-1} & A^{spl}_0\end{pmatrix}\in\mathcal{M}_{(N_r+3)\times N_\theta,(N_r+1)\times N_\theta}(\mathbb{R})$$

$$A^{contr}_{\rho_j}=\begin{pmatrix}A^{contr}_{\rho_j,0} & A^{contr}_{\rho_j,1} & \dots & A^{contr}_{\rho_j,N_\theta-1}\\ A^{contr}_{\rho_j,N_\theta-1} & \ddots & \ddots & \vdots\\ \vdots & \ddots & \ddots & A^{contr}_{\rho_j,1}\\ A^{contr}_{\rho_j,1} & \dots & A^{contr}_{\rho_j,N_\theta-1} & A^{contr}_{\rho_j,0}\end{pmatrix}\in\mathcal{M}_{(N_r+1)\times N_\theta,(N_r+3)\times N_\theta}(\mathbb{R})$$

où $A^{spl}_i\in\mathcal{M}_{N_r+3,N_r+1}(\mathbb{R})$ et $A^{contr}_{\rho_j,i}\in\mathcal{M}_{N_r+1,N_r+3}(\mathbb{R})$ pour $i=0,...,N_\theta-1$. Ces matrices sont alors diagonalisables dans la base de Fourier vectoriel :

$$A^{spl}=U_{N_r+3}D^{spl}U^\star_{N_r+1},\qquad A^{contr}_{\rho_j}=U_{N_r+1}D^{contr}_{\rho_j}U^\star_{N_r+3}$$

où

$$D_{spl}=\begin{pmatrix}D^{spl}_0 & & \\ & \ddots & \\ & & D^{spl}_{N_\theta-1}\end{pmatrix}\qquad D^{contr}_{\rho_j}=\begin{pmatrix}D^{contr}_{\rho_j,0} & & \\ & \ddots & \\ & & D^{contr}_{\rho_j,N_\theta-1}\end{pmatrix}$$

avec

$$D_m^{spl} = \sum_{k=0}^{N_\theta-1} A_k^{spl} e^{\frac{-2i\pi km}{N_\theta}}, \qquad D_{\rho_j,m}^{contr} = \sum_{k=0}^{N_\theta-1} A_{\rho_j,k}^{contr} e^{\frac{-2i\pi km}{N_\theta}}$$

et

$$U_n = \begin{pmatrix} U_{n,0,0} & \dots & U_{n,0,N_\theta-1} \\ \vdots & \ddots & \vdots \\ U_{n,N_\theta-1,0} & \dots & U_{n,N_\theta-1,N_\theta-1} \end{pmatrix}, \qquad U_{n,k,\ell} = \frac{1}{\sqrt{N_\theta}} e^{\frac{2i\pi k\ell}{N_\theta}} I_n$$

où I_n est la matrice identité de taille $n \times n$.

L'intérêt de la diagonalisation dans la base de Fourier est que le calcul de la gyro-moyenne de Φ par le produit $G_{\rho_j}\Phi$ se fait de manière rapide avec les FFT :

1. Passage dans la base de Fourier par FFT(Φ).
2. Calcul du produit $G_{\rho_j}\Phi$ dans la base de Fourier.
3. Passage dans l'espace réel par FFT^{-1} du résultat précédent.

L'utilisation du maillage polaire et de la FFT permettent de faire des calculs plus rapidement et constitue une base pour des travaux en géométrie plus complexe.

9.3 Solveur par approximation de Padé

Dans cette section, nous présentons une méthode de résolution de l'équation de quasi-neutralité basée sur l'approximation de Padé de la fonction de Bessel utilisée afin de calculer le terme

$$\tilde{\Phi} = \frac{1}{T_i} \int_{\mathbb{R}^+} \mathcal{J}^2_{\sqrt{2\mu}}(\Phi) e^{-\mu/T_i} d\mu = \int_{\mathbb{R}^+} \mathcal{J}^2_{\sqrt{2\mu T_i}}(\Phi) e^{-\mu} d\mu$$

de l'équation de quasi-neutralité (9.1.1). Une telle méthode a déjà été décrite dans le chapitre précédent. Dans ce chapitre, nous avions montré qu'en passant dans l'espace de Fourier (voir 8.1.2) :

$$\widehat{\mathcal{J}_\rho(\Phi)}(\mathbf{k}) = J_0(|\mathbf{k}|\rho)\widehat{\Phi}(\mathbf{k}).$$

Ainsi, nous obtenons :

$$\widehat{\tilde{\Phi}}(\mathbf{k}) = \Gamma_0(|\mathbf{k}|^2 T_i)\widehat{\Phi}(\mathbf{k})$$

où la fonction Γ_0 est définie par :

$$\Gamma_0(k^2) := \int_{\mathbb{R}^+} \exp(-x^2/2) J_0^2(kx) x dx.$$

Nous utilisons alors les développements de Padé et de Taylor de la fonction de Bessel :

$$J_0(kx) \approx \frac{2}{1 + (kx)^2/4} \approx 1 - \frac{(kx)^2}{4}$$

ce qui donne le développement de Taylor de la fonction Γ_0 :

$$\begin{aligned}\Gamma_0(|\mathbf{k}|^2 T_i) &= \int_{\mathbb{R}^+} \exp(-x^2/2) J_0^2(|\mathbf{k}|\sqrt{T_i}x) x dx \\ &= \int_{\mathbb{R}^+} \exp(-x^2/2) x dx - \frac{|\mathbf{k}|^2 T_i}{2} \int_{\mathbb{R}^+} \exp(-x^2/2) x^3 dx + \mathcal{O}(|\mathbf{k}|^3) \\ &= 1 - |\mathbf{k}|^2 T_i + \mathcal{O}(|\mathbf{k}|^3).\end{aligned}$$

En utilisant ce développement de la fonction Γ_0, nous aboutissons à l'équation (voir [102, 101, 104]) :

$$-\nabla_\perp \cdot (n_i \nabla_\perp \Phi) + \frac{n_0}{T_e}(\Phi - \langle \Phi \rangle) = \bar{n}_i - n_0.$$

Une linéarisation du terme de diffusion en supposant que $n_i(\mathbf{x}) \approx n_0(r)$ (voir [80, 102]) conduit à :

$$-\nabla_\perp \cdot (n_0 \nabla_\perp \Phi) + \frac{n_0}{T_e}(\Phi - \langle \Phi \rangle) = \bar{n}_i - n_0$$

et finalement à

$$-\left(\partial_r^2 \Phi + \left(\frac{1}{r} + \frac{\partial_r n_0(r)}{n_0(r)}\right)\partial_r \Phi + \frac{1}{r^2}\partial_\theta^2 \Phi\right) + \frac{1}{T_e}(\Phi - \langle \Phi \rangle) = \frac{1}{n_0}(\bar{n}_i - n_0). \tag{9.3.1}$$

La résolution de cette équation de Poisson s'effectue en passant dans l'espace de Fourier en θ et par différences finies en r comme cela a été détaillé dans la section 8.2.

Modèle simplifié

Dans la suite, nous considérons un cas simplifié de l'équation de quasi-neutralité (9.1.1) où les intégrales en μ sont remplacées par l'évaluation en une valeur de $\mu = \mu_0$:

$$\frac{n_0}{T_i}(\Phi - \mathcal{J}^2_{\sqrt{2\mu_0 T_i}}(\Phi)) + \frac{n_0}{T_e}(\Phi - \langle \Phi \rangle) = \int \mathcal{J}_{\sqrt{2\mu_0}}(f)(\mathbf{x}, v_z) dv_z - n_0. \tag{9.3.2}$$

Dans l'espace de Fourier,

$$\widehat{[\Phi - \mathcal{J}^2_{\sqrt{2\mu_0 T_i}}(\Phi)]}(\mathbf{k}) = \left(1 - J_0^2(|\mathbf{k}|\sqrt{2\mu_0 T_i})\right)\widehat{\Phi}(\mathbf{k}).$$

Le développement de Taylor du facteur précédent donne

$$\begin{aligned}1 - J_0^2(|\mathbf{k}|\sqrt{2\mu_0 T_i}) &= 1 - \left(1 - \frac{(|\mathbf{k}|\sqrt{2\mu_0 T_i})^2}{2} + \mathcal{O}(|\mathbf{k}|^4)\right) \\ &= |\mathbf{k}|^2 \mu_0 T_i + \mathcal{O}(|\mathbf{k}|^4).\end{aligned}$$

On aboutit alors à l'équation de Poisson suivante :

$$-\mu_0 \Delta_\perp \Phi + \frac{1}{T_e}(\Phi - \langle \Phi \rangle) = \frac{1}{n_0}\left(\int \mathcal{J}_{\sqrt{2\mu_0}}(f)(\mathbf{x}, v_z) dv_z - n_0\right).$$

La Figure 9.1 donne les représentations de $x \mapsto (1 - J_0^2(x))^{-1}$ et de son développement de Padé $x \mapsto 2/x^2$. Nous observons que lorsque x tend vers 0, les deux fonctions coïncident, ce qui montre que lorsque $|\mathbf{k}|\sqrt{2\mu_0 T_i}$ est petit, la solution obtenue avec la méthode de Padé appliqué à un μ est proche de la solution au niveau continu. Lorsque x augmente, l'approximation de Padé est sous la fonction $x \mapsto (1 - J_0^2(x))^{-1}$, ce qui justifiera dans les résultats numériques que le taux d'instabilité de la méthode de Padé appliquée à un μ soit plus faible que celui de la méthode par interpolation.

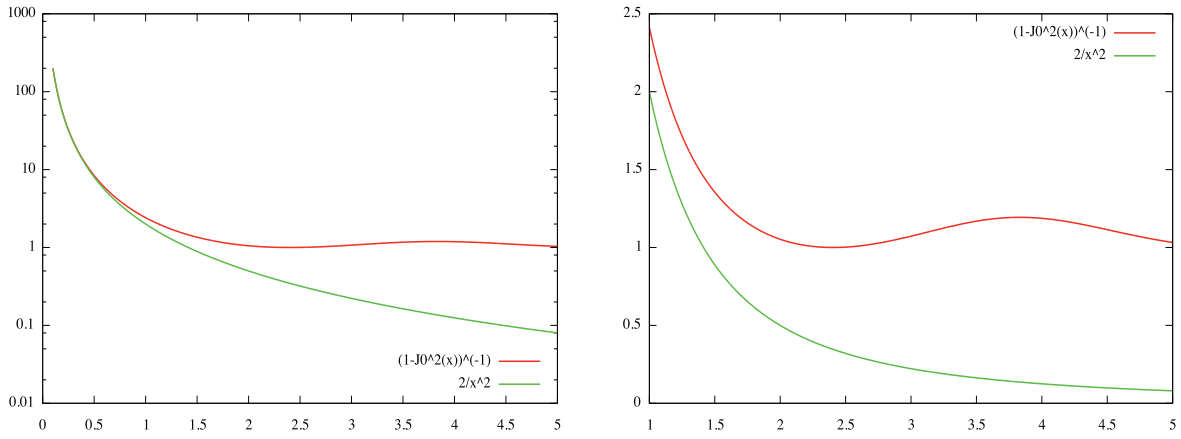


FIGURE 9.1 – Représentations de $x \mapsto (1 - J_0^2(x))^{-1}$ (en rouge) et de son développement de Padé $x \mapsto 2/x^2$ (en vert).

9.4 Résultats Numériques

9.4.1 Cas test analytiques

L'opérateur de quasi-neutralité est diagonalisable dans la base de Fourier-Bessel. Nous pouvons donc réutiliser les solutions analytiques développées dans la sous-section 8.4.1. Considérons ainsi la fonction

$$\Phi(r,\theta) = \left(J_m(\gamma_{m,\ell}) Y_m\left(r\frac{\gamma_{m,\ell}}{r_{\max}} \right) - Y_m(\gamma_{m,\ell}) J_m\left(r\frac{\gamma_{m,\ell}}{r_{\max}} \right) \right) e^{im\theta} \tag{9.4.1}$$

où $\gamma_{m,\ell}$ est le $\ell^{ème}$ zéro de l'application

$$y \mapsto J_m(y) Y_m\left(y\frac{r_{\min}}{r_{\max}} \right) - Y_m(y) J_m\left(y\frac{r_{\min}}{r_{\max}} \right).$$

Alors Φ vérifie des conditions de Dirichlet homogène aux bords :

$$\Phi(r_{\min},\theta) = 0, \quad \Phi(r_{\max},\theta) = 0, \quad 0 \leq \theta < 2\pi,$$

et sa gyromoyenne vaut

$$\mathcal{J}_{\sqrt{2\mu}}(\Phi)(r_0,\theta_0) = J_0\left(\sqrt{2\mu}\frac{\gamma_{m,\ell}}{r_{\max}} \right) \Phi(r_0,\theta_0).$$

— Dans le cas de l'opérateur de quasi-neutralité avec l'intégrale en μ sans considérer le terme $\langle\Phi\rangle$ (cas sans zonal flow), nous obtenons :

$$\int_0^{+\infty} (\Phi - \mathcal{J}^2_{\sqrt{2\mu}}(\Phi)) e^{-\mu} d\mu + \lambda\Phi = \left(1 + \lambda - \Gamma_0\left(\frac{\gamma^2_{m,\ell}}{r^2_{\max}} \right) \right) \Phi.$$

en considérant $T_i = 1$ et pour un λ réel quelconque.

— Dans le cas de l'opérateur de quasi-neutralité avec un μ et sans considérer le terme $\langle\Phi\rangle$ (cas sans zonal flow), nous obtenons :

$$(\Phi - \mathcal{J}^2_{\sqrt{2\mu_i}}(\Phi)) + \lambda\Phi = \left(1 + \lambda - J_0^2\left(\sqrt{2\mu}\frac{\gamma_{m,\ell}}{r_{\max}}\right)\right)\Phi.$$

Nous avons comparé la méthode par Padé à la méthode par interpolation sur cette solution de référence dans le cas de l'opérateur de quasi-neutralité avec l'intégrale en μ. Nous avons choisi les paramètres suivants : $N_r = N_\theta = 64$, $\lambda = 1$, $\ell = 1$. Pour la méthode par interpolation, nous considérons $\mu_{\max} = 35$, $N_\mu = 100$ (quadrature par Gauss ; 10 points de Gauss sont utilisés) et $N = 128$. Nous avons testé les cas $m = 1$ et $m = 5$. La Figure 9.2 représente, pour $m = 1$, la solution exacte de l'équation de quasi-neutralité avec l'intégrale en μ (surface rouge) et les solutions obtenues avec la méthode de Padé (surface verte en haut) et la méthode par interpolation (surface verte en bas). Il apparaît que la méthode par interpolation crée des oscillations en $r_{\min}$ et $r_{\max}$ contrairement à la méthode de Padé. Ceci est confirmé par la Table 9.1 qui donne la norme L^2 de l'erreur relative :

$$\left\| \frac{\Phi_{schema} - \Phi}{\Phi} \right\|_2$$

pour Φ_{schema} obtenu avec les différentes méthodes de résolution de l'équation de quasi-neutralité (méthode par interpolation ou Padé) avec $m = 1$. L'erreur pour la méthode par interpolation diminue lorsque nous restreignons l'intervalle en plaçant deux zones buffer $[r_{\min}, r_{\min} + \alpha]$ et $[r_{\max} - \alpha, r_{\max}]$ sur lesquelles l'erreur n'est pas calculée. Dans la Table 9.1, la taille α des zones buffer est exprimée en nombre de cellules. En restreignant l'intervalle de manière significative dans le cas de la méthode par interpolation afin de ne pas être perturbé par les oscillations aux bords du domaine, nous retrouvons une erreur comparable à celle de la méthode de Padé. Le choix de $m = 1$ correspond au domaine de validité de Padé, ce qui explique que l'erreur soit assez faible. Cependant, lorsque m augmente ($m = 5$ pour la Figure 9.3 et la Table 9.2), la méthode par interpolation donne clairement de meilleurs résultats que la méthode par Padé à l'intérieur du domaine bien que la méthode par interpolation présente à nouveau des oscillations au bord.

La Table 9.3 présente les temps d'exécution des méthodes par Padé et par interpolation appliquées 1000 fois sur la fonction analytique ci-dessus. Cette table montre, que pour la méthode par interpolation, le temps de précalcul est proportionnel au nombre de μ. De plus, la méthode par Padé est plus rapide que la méthode par interpolation d'un facteur 10, ce qui est en accord avec les tests de performance réalisés dans le cadre de la gyromoyenne (chapitre 8) où nous avions également trouvé un facteur proche de 10 entre Padé et la méthode par interpolation.

9.4.2 Application aux simulations gyrocinétiques

Nous présentons ici uniquement des résultats numériques dans le cas de l'équation de quasi-neutralité avec un μ (équation 9.3.2). Considérer le cas avec un μ est justifié par le fait que le cas général nécessite un cas $5D$ avec une condition initiale $5D$ ce qui est très coûteux en temps de calcul. De plus, ces cas test sont plus compliqués car ils font intervenir plus de physique ; nous cherchons à valider et donc à utiliser un cas un peu plus simple. Le cas

Taille de la zone buffer	0	1	2	3	4	5	10	20
Interpolation	0.28	0.28	0.13	7.10^{-2}	5.10^{-2}	3.10^{-2}	8.10^{-3}	1.10^{-3}
Padé	1.10^{-3}	1.10^{-3}	1.10^{-3}	1.10^{-3}	1.10^{-3}	1.10^{-3}	1.10^{-3}	1.10^{-3}

TABLE 9.1 – Norme L^2 de l'erreur relative avec une zone buffer en r de taille variable (en nombre de cellules) pour $m = 1$. Paramètres : $N_r = N_\theta = 64$, $\lambda = 1$, $\ell = 1$. Méthode par interpolation : $\mu_{\max} = 35$, $N = 128$, $N_\mu = 100$.

Taille de la zone buffer	0	1	2	3	4	5	10	20
Interpolation	0.20	0.18	8.10^{-2}	5.10^{-2}	3.10^{-2}	2.10^{-2}	5.10^{-3}	7.10^{-4}
Padé	3.10^{-2}	3.10^{-2}	3.10^{-2}	3.10^{-2}	3.10^{-2}	3.10^{-2}	3.10^{-2}	3.10^{-2}

TABLE 9.2 – Norme L^2 de l'erreur relative avec une zone buffer en r de taille variable (en nombre de cellules) pour $m = 5$. Paramètres : $N_r = N_\theta = 64$, $\lambda = 1$, $\ell = 1$. Méthode par interpolation : $\mu_{\max} = 35$, $N = 128$, $N_\mu = 100$.

	Padé	Méthode par interpolation					
Nombre de μ pour $\tilde{\Phi}$	-	1	2	4	8	32	128
Temps d'initialisation (s.)	0.005	2.6	4.4	8.2	16.3	64.0	256
Temps d'exécution (s.)	2.4	27	26	27	26	26	27

TABLE 9.3 – 1000 itérations, 64×64 cellules, irma-hpc2, $N = 1024$.

de l'équation de quasi-neutralité avec l'intégrale en μ (équation 9.1.1) sera traité dans un travail ultérieur.

L'équation satisfaite par la fonction de distribution des ions $f(t, r, \theta, z, v)$ suivant le mouvement du centre guide se lit :

$$\partial_t f - \left(\frac{\partial_\theta \mathcal{J}_{\sqrt{2\mu_0}}\Phi}{r}\right)\partial_r f + \left(\frac{\partial_r \mathcal{J}_{\sqrt{2\mu_0}}\Phi}{r}\right)\partial_\theta f + v\partial_z f - \left(\partial_z \mathcal{J}_{\sqrt{2\mu_0}}\Phi\right)\partial_v f = 0. \tag{9.4.2}$$

pour $(r, \theta, z, v, \mu) \in [r_{\min}, r_{\max}] \times [0, 2\pi] \times [0, L] \times [-v_{\max}, v_{\max}]$. Cette équation sera couplée soit avec l'équation de quasi-neutralité :

$$\frac{n_0}{T_i}(\Phi - \mathcal{J}^2_{\sqrt{2\mu_0 T_i}}(\Phi)) + \frac{n_0}{T_e}(\Phi - \langle\Phi\rangle) = \int \mathcal{J}_{\sqrt{2\mu_0}}(f)(\mathbf{x}, v_z)dv_z - n_0.$$

résolue par la méthode par interpolation, soit avec l'équation de Poisson :

$$-\mu_0\Delta_\perp\Phi + \frac{1}{T_e}(\Phi - \langle\Phi\rangle) = \frac{1}{n_0}\left(\int \mathcal{J}_{\sqrt{2\mu_0}}(f)(\mathbf{x}, v_z)dv_z - n_0\right).$$

dans le cas de la méthode par Padé pour un μ. La résolution de cette équation de Poisson s'effectue en passant dans l'espace de Fourier en θ et par différences finies en r comme cela

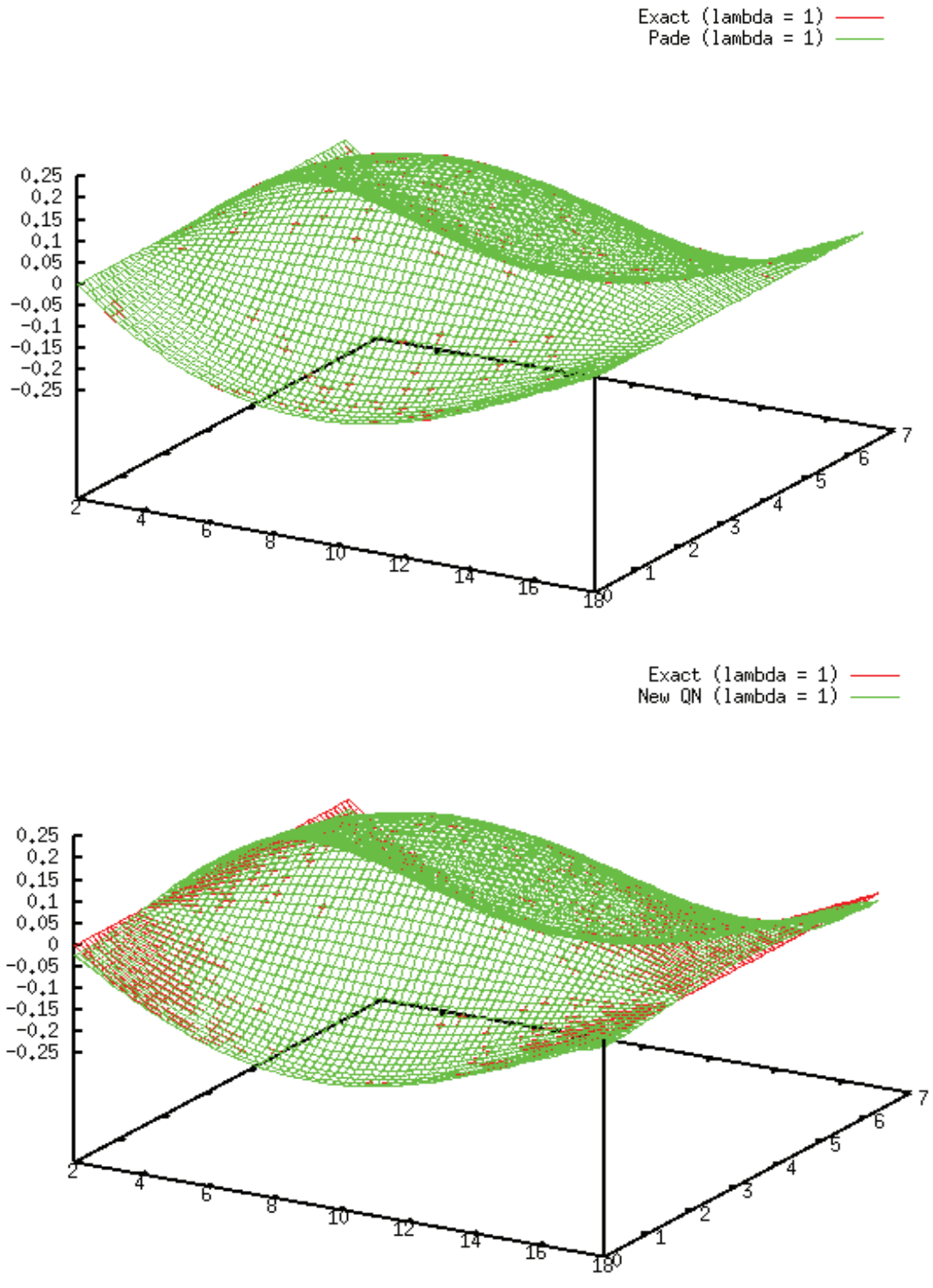


FIGURE 9.2 – Comparaison entre la solution analytique de l'équation de quasi-neutralité (en rouge) et la solution obtenue avec la méthode par Padé (en vert en haut) et celle obtenue avec la méthode par interpolation (en vert en bas) pour $m = 1$.

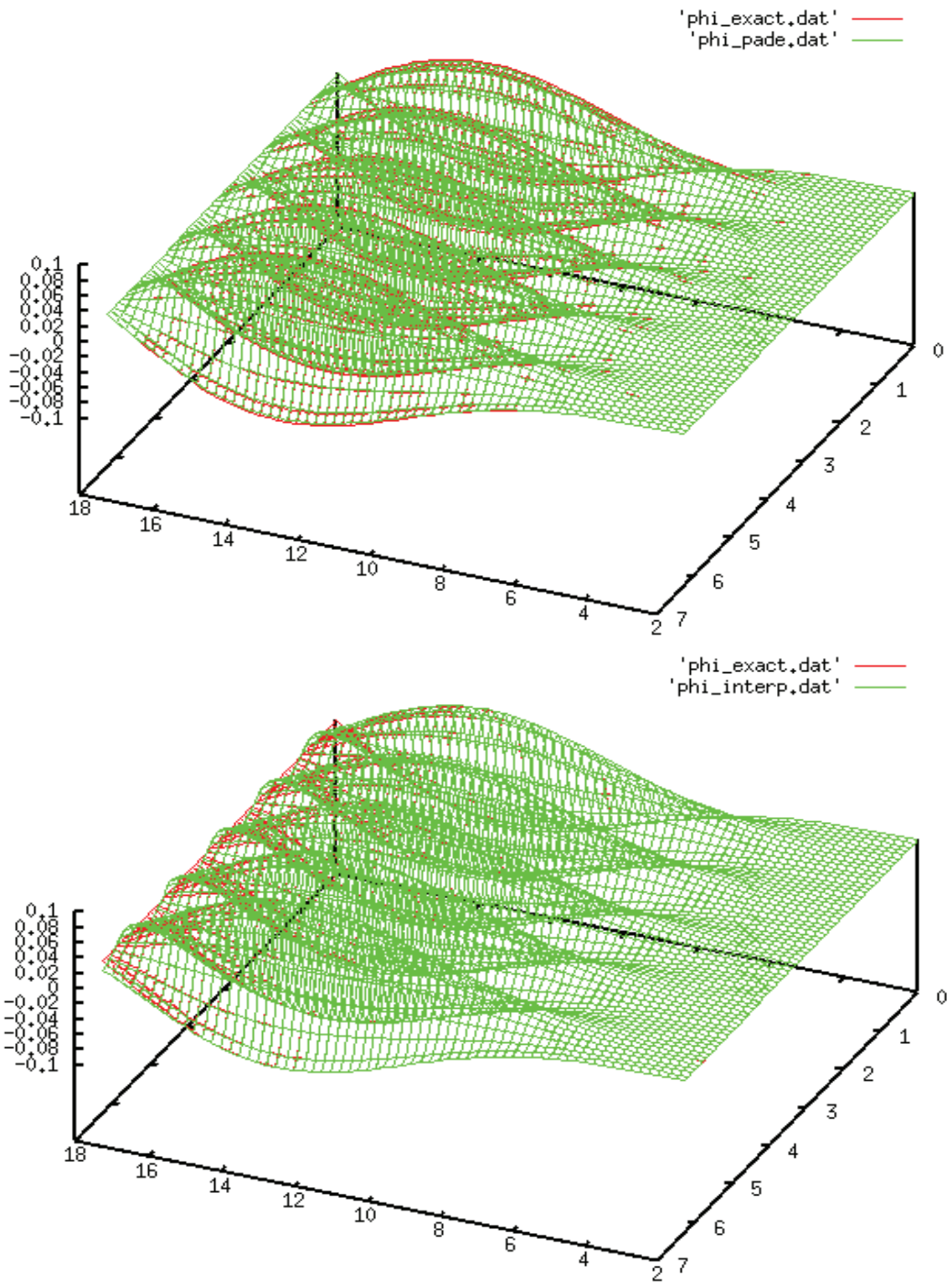


FIGURE 9.3 – Comparaison entre la solution analytique de l'équation de quasi-neutralité (en rouge) et la solution obtenue avec la méthode par Padé (en vert en haut) et celle obtenue avec la méthode par interpolation (en vert en bas) pour $m = 5$.

a été détaillé dans la section 8.2.

Pour traiter ce système d'équations, nous avons utilisé la plate-forme SELALIB [67]. L'équation de Vlasov est résolue avec une méthode semi-Lagrangienne classique avec interpolation par splines cubiques, ainsi qu'une méthode prédicteur-correcteur et l'algorithme de Verlet pour les caractéristiques (voir [86, 77] pour plus de détails). Dans les simulations, nous ne considérerons pas le terme $\langle\Phi\rangle$ (cas sans "zonal flow"). La fonction de distribution initiale se lit :

$$f(0,r,\theta,z,v) = f_{eq}(r,v) \times \left(1+\varepsilon\exp\left(-\frac{(r-r_p)^2}{\delta r}\right)\cos\left(\frac{2\pi n}{L}z+m\theta\right)\right)$$

où la fonction d'équilibre f_{eq} vaut

$$f_{eq}(r,v) = \frac{n_0(r)\exp\left(-\frac{v^2}{2T_i(r)}\right)}{(2\pi T_i(r))^{1/2}}.$$

Les profils T_i, T_e et n_0 sont donnés par :

$$\mathcal{P}(r) = C_{\mathcal{P}}\exp\left(-\kappa_{\mathcal{P}}\delta r_{\mathcal{P}}\tanh\left(\frac{r-r_p}{\delta r_{\mathcal{P}}}\right)\right)$$

où $\mathcal{P}\in\{T_i,T_e,n_0\}$, $C_{T_i}=C_{T_e}=1$ et

$$C_{n_0} = \frac{r_{\max}-r_{\min}}{\int_{r_{\max}}^{r_{\min}}\exp\left(-\kappa_{n_0}\delta r_{n_0}\tanh\left(\frac{r-r_p}{\delta r_{n_0}}\right)\right)dr}.$$

Nous considérons les paramètres de [86] [Medium case] :

$$\begin{aligned} r_{\min}=0.1, r_{\max}=14.5, v_{\max}=7.32, \kappa_{n_0}=0.055,\\ \kappa_{T_i}=\kappa_{T_e}=0.27586, \delta r_{T_i}=\delta r_{T_i}=\frac{\delta r_{n_0}}{2}=1.45,\\ \varepsilon=10^{-6}, n=1, m=5,\\ L=1506.759067, r_p=\frac{r_{\min}+r_{\max}}{2}, \delta_r=\frac{4\delta r_{n_0}}{\delta r_{T_i}}. \end{aligned}$$

Les résultats numériques sont donnés dans les Fig. 9.4 – 9.7. Nous considérons ici $N=1024$ pour la méthode par interpolation. La méthode par interpolation est utilisée pour le membre de droite de l'équation de quasi-neutralité même lorsque Padé est utilisé pour l'opérateur de quasi-neutralité afin de voir l'influence des différentes méthodes uniquement sur l'opérateur de quasi-neutralité. Sur la Fig. 9.4, nous traçons une coupe poloïdale $f(r,\theta,0,0)$ ($\mu=0.1$ au temps $T=3000$ et $\mu=0.2$ au temps $T=3500$) pour Padé et la méthode par interpolation. Nous observons globalement les mêmes structures à la différence que la méthode par interpolation semble générer davantage d'oscillations. Sur la Fig. 9.5, nous augmentons la valeur de $\mu=0.7$. Nous observons que la méthode par interpolation conduit à davantage de structures sans qu'il n'y ait de décalage entre les résultats des deux

méthodes. Nous avions déjà observé ce phénomène dans le cas de la gyromoyenne simple (Fig. 8.8 et 8.9). Cependant, il n'y a globalement pas beaucoup de différences entre les deux méthodes car les taux d'instabilité sont encore proches quand $\mu = 0.7$.
Sur la Figure 9.6, nous considérons $\mu = 1$. Pour cette valeur de μ, nous constatons des différences entre les deux méthodes : les structures sont déjà bien plus développées pour le solveur par interpolation, ce qui sera confirmé par l'analyse des taux d'instabilité (Fig. 9.7). De plus, nous observons la convergence en maillage.
La Figure 9.7 montre l'évolution temporelle de $\int_{r_{\min}}^{r_{\max}} \int_0^{2\pi} \Phi(r,\theta,0) r dr d\theta$ dans la phase linéaire puis non linéaire. Nous remarquons d'abord que pour $\mu = 0$, les deux méthodes sont instables et explosent très rapidement (courbes épaisses verte et rouge). Ensuite, plus μ augmente ($\mu = 0.1, 0.2, 0.5, 0.7, 1$), plus le taux d'instabilité diminue pour les deux méthodes. Pour un μ donné, la méthode par Padé semble avoir un taux d'instabilité plus bas que pour la méthode par interpolation. En effet, dans la relation de dispersion (voir Appendice G), les facteurs qui interviennent pour la méthode par Padé et pour la méthode par interpolation sont ordonnés (voir Fig. 9.1) ce qui justifie que le taux d'instabilité pour la méthode par Padé est plus faible.

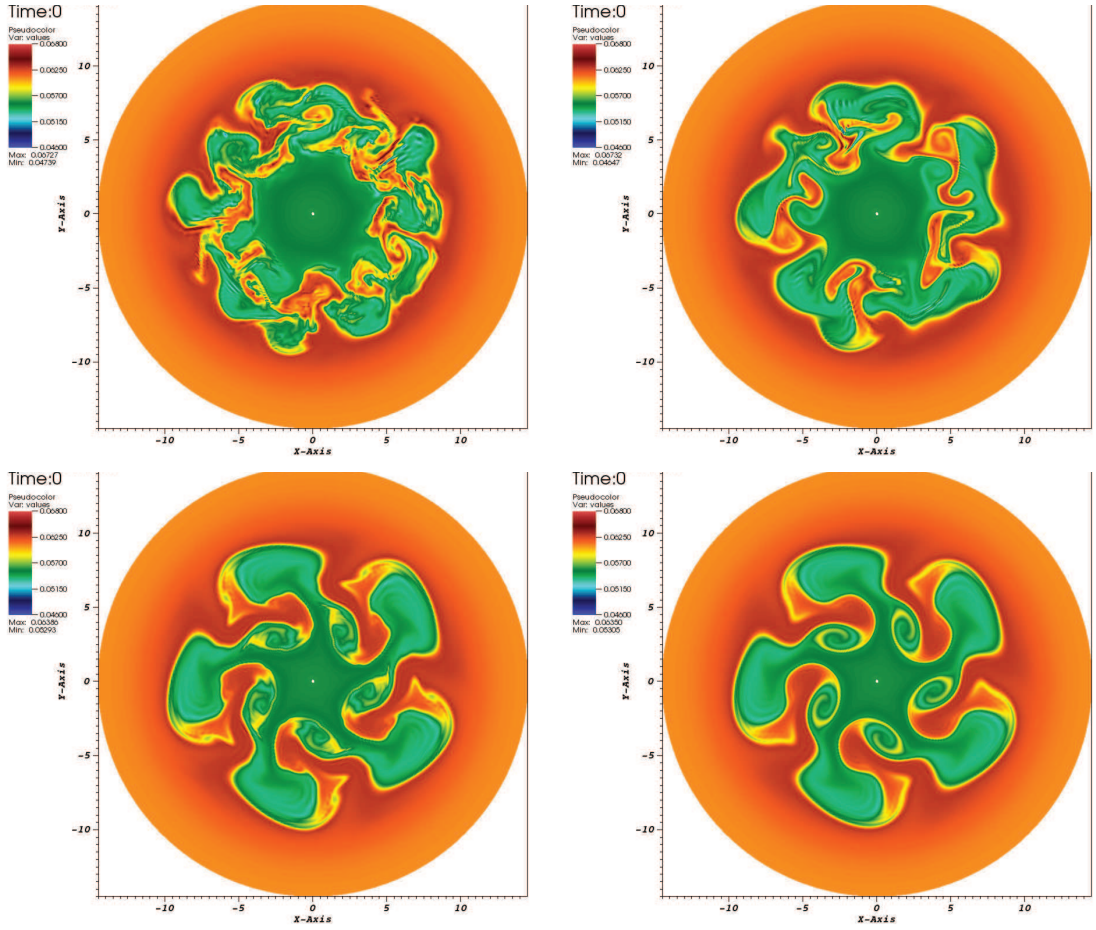


FIGURE 9.4 – Coupe poloïdale $f(r, \theta, z = 0, v_{\|} = 0)$ pour $128 \times 256 \times 32 \times 128, \Delta t = 2$. En haut $\mu = 0.1$ au temps $T = 3000$ (de gauche à droite) : avec solveur QN par interpolation et puis Padé ; en bas $\mu = 0.2$ au temps $T = 3500$ (de gauche à droite) : avec solveur QN par interpolation et puis Padé.

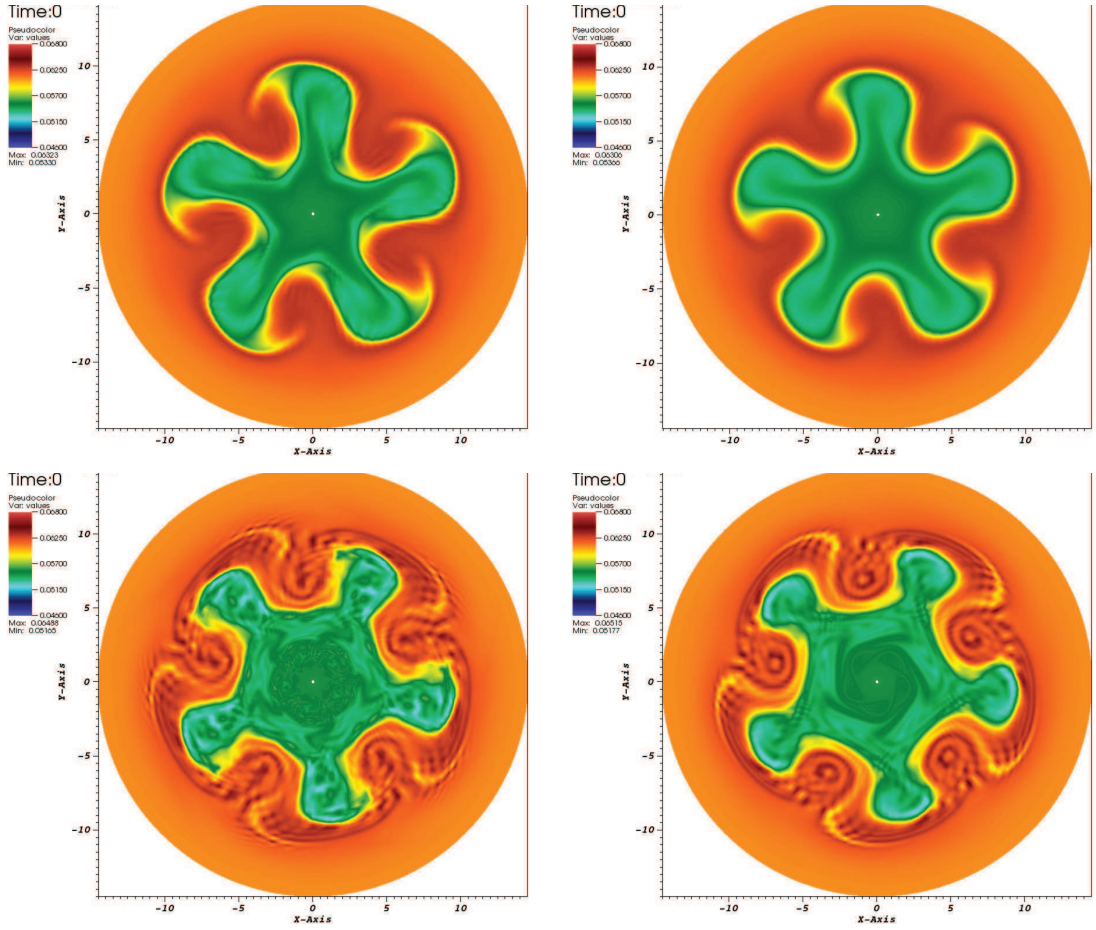


FIGURE 9.5 – Coupe poloïdale $f(r, \theta, z = 0, v_{\|} = 0)$ sur un maillage $128 \times 128 \times 32 \times 128$, $\Delta t = 2$, $\mu = 0.7$, aux temps $T = 5000$ (en haut) et $T = 7000$ (en bas), avec solveur QN par interpolation (à gauche) et Padé (à droite).

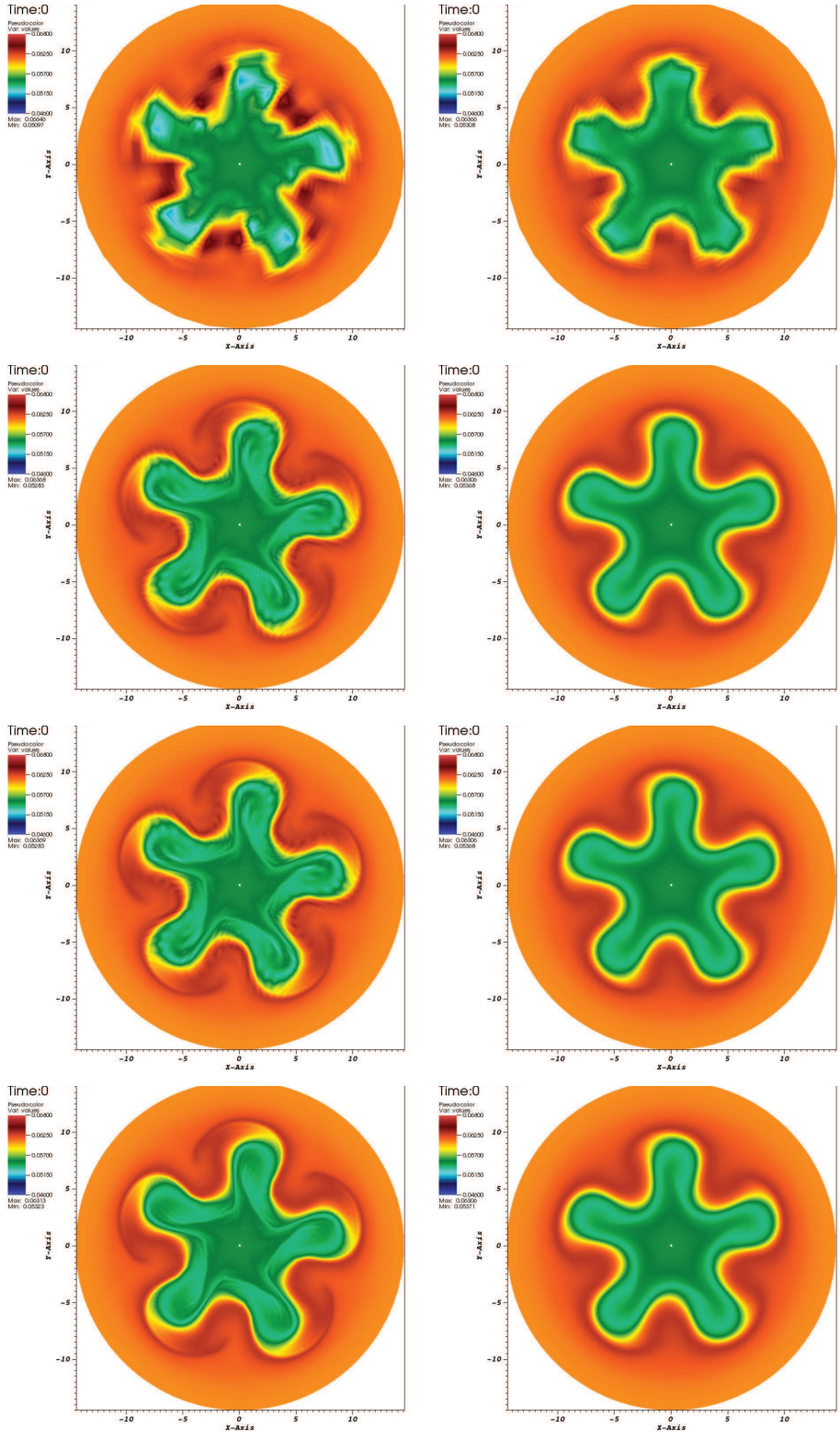


FIGURE 9.6 – Coupe poloïdale $f(r, \theta, z = 0, v_{\|} = 0)$ avec $\Delta t = 2$, $\mu = 1$, au temps $T = 7000$ avec solveur QN par interpolation (à gauche) et Padé (à droite). Maillages (de haut en bas) : $32 \times 32 \times 32 \times 64$; $64 \times 128 \times 32 \times 128$; $128 \times 128 \times 32 \times 128$ et $128 \times 256 \times 32 \times 128$.

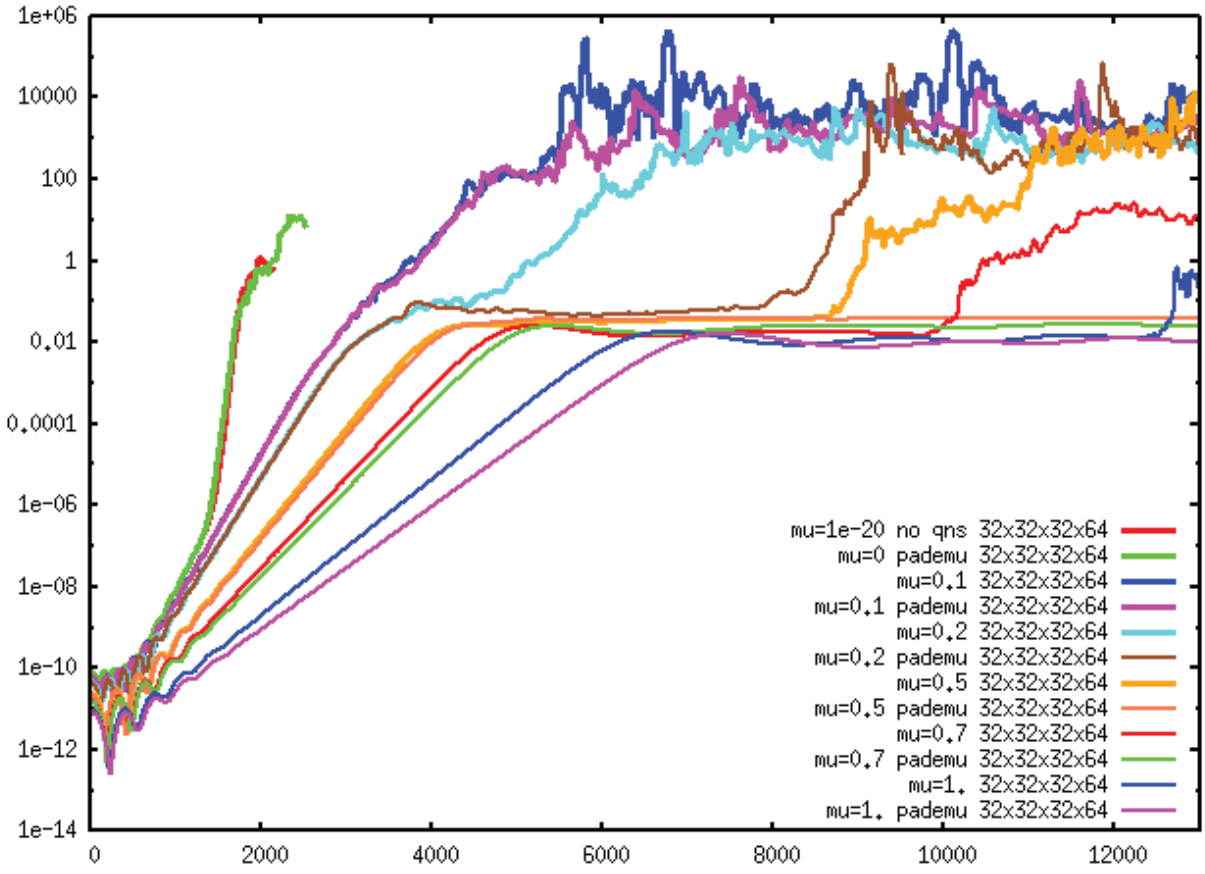


FIGURE 9.7 – Evolution en temps de $\int_{r_{\min}}^{r_{\max}} \int_0^{2\pi} \Phi(r,\theta,0) r dr d\theta$ sur un maillage $32\times32\times32\times64$, avec Padé pour un μ (pademu dans la légende) et avec la méthode par interpolation pour différents μ (0, 0.1, 0.2, 0.5, 0.7, 1).

Conclusion générale & Perspectives

Dans ce manuscrit, nous avons proposé et analysé différentes méthodes numériques pour la résolution de l'équation de Vlasov.

La première partie concerne la résolution de l'équation d'advection linéaire qui intervient de manière récurrente dans les méthodes semi-Lagrangiennes. Nous avons établi les équations équivalentes de plusieurs schémas résolvant cette équation. Nous avons montré que l'approche par les équations équivalentes fonctionne en considérant suffisamment de termes. Cette approche permet de prévoir les propriétés de diffusion et de dispersion des schémas et de les comparer entre eux. Dans le chapitre 3, nous nous sommes intéressés plus particulièrement au schéma Galerkin Discontinu Semi-Lagrangien pour lequel nous avons montré numériquement et formellement une propriété de superconvergence pour les petits degrés.

La poursuite des investigations permettant d'établir la preuve générale de ce résultat pour un degré quelconque et l'adaptation de cette preuve lorsque le déplacement tend vers 0 constituent les perspectives de travail de cette partie.

La seconde partie expose des méthodes de résolution du système de Vlasov-Poisson. Des schémas de volumes finis ont été étudiés et comparés dans le chapitre 4. Nous avons montré un lien entre les méthodes de volumes finis et les méthodes semi-Lagrangiennes dans le cadre de l'équation d'advection. Lorsque le pas de temps Δt tend vers zéro, les méthodes semi-Lagrangiennes récupèrent des caractéristiques des méthodes de volumes finis. Dans le chapitre 5, nous avons détaillé une implémentation sur GPU d'une méthode de résolution du système de Vlasov-Poisson. Nous avons montré que cette approche fonctionne et conduit à d'importants speed-ups. Le chapitre 6 était consacré à l'implémentation du schéma Galerkin Discontinu semi-Lagrangien sur maillage non uniforme. Nous avons validé cette méthode par comparaison aux splines cubiques non uniformes préexistantes. Ainsi, il est possible de réduire le nombre de points ; ceci est intéressant dans le cas test des ondes KEEN et est encourageant pour les futures simulations $2D \times 2D$.

Concernant le code sur GPU, nous espérons monter en dimension en implémentant une version en quatre dimensions (2 en x et 2 en v) de ce code puis inclure des collisions. Un travail permettant l'accélération du code non uniforme SLDG est également envisagé avec à terme une intégration dans la librairie SELALIB. Pour toutes ces méthodes, nous aimerions faire de la décomposition de domaine dynamique (maillages fins et grossiers évoluant au cours du temps) pour appliquer cette approche à d'autres contextes.

La troisième et dernière partie présente des méthodes numériques pour la résolution du modèle gyrocinétique. Dans le chapitre 7, nous avons présenté un opérateur d'interpolation

de type Hermite que nous avons utilisé dans le cadre de la résolution numérique du modèle Drift-Kinetic 4D. Nous avons montré que dans le cas d'une reconstruction des dérivées décentrées, le schéma est plus diffusif et crée moins d'oscillations numériques pour de petits pas de temps. Une méthode de calcul de l'opérateur de gyromoyenne en géométrie polaire basée sur l'interpolation a été présentée et validée dans le chapitre 8. Nous avons comparé cette nouvelle méthode à la méthode classique par Padé sur des cas test analytiques ainsi que sur des simulations gyrocinétiques. Nous avons montré que l'introduction de l'opération de gyromoyenne diminue le taux d'instabilité. Ce taux diminue davantage en considérant l'opérateur de gyromoyenne par interpolation par rapport à la méthode par Padé. Dans le dernier chapitre, nous avons utilisé cette méthode de calcul de gyromoyenne par interpolation pour résoudre l'équation de quasi-neutralité. Des comparaisons sont faites sur des cas test analytiques ainsi que sur des simulations gyrocinétiques avec la méthode classique de Padé. On montre que la méthode par interpolation donne de meilleurs résultats pour de hauts modes en theta malgré l'apparition d'oscillations aux bords.

Dans un travail futur, il serait intéressant d'adapter l'opérateur de gyromoyenne à une géométrie plus complexe que la géométrie polaire. Concernant l'équation de quasi-neutralité, nous aimerions considérer l'équation de quasi-neutralité avec l'intégrale en μ en discutant avec des physiciens pour l'interprétation des résultats. La méthode de résolution de l'équation de quasi-neutralité par interpolation pourrait également être améliorée en définissant mieux l'opérateur aux bords en r afin d'éviter les oscillations. Un travail sur d'optimisation du code est également souhaitable afin de rendre la méthode par interpolation plus compétitive par rapport à la méthode de Padé en terme de temps de calcul.

Appendices

E Polynômes de Tchebychev

Définition E.1. *Soit $n \in \mathbb{N}$. Il existe un unique polynôme T_n tel que*

$$\forall \theta \in \mathbb{R},\ T_n(\cos(\theta)) = \cos(n\theta).$$

Les T_n ($n \in \mathbb{N}$) sont appelés polynômes de Tchebychev de première espèce.

Exemple E.2. *Les premiers polynômes de Tchebychev de première espèce valent :*
$T_0(X) = 1$,
$T_1(X) = X$,
$T_2(X) = 2X^2 - 1$,
$T_3(X) = 4X^3 - 3X$.

Définition E.3. *Soit $n \in \mathbb{N}$. Il existe un unique polynôme U_n tel que*

$$\forall \theta \in \mathbb{R},\ \sin(\theta) \times U_n(\cos(\theta)) = \sin((n+1)\theta).$$

Les U_n ($n \in \mathbb{N}$) sont appelés polynômes de Tchebychev de seconde espèce.

Exemple E.4. *Les premiers polynômes de Tchebychev de seconde espèce valent :*
$U_0(X) = 1$,
$U_1(X) = 2X$,
$U_2(X) = 4X^2 - 1$,
$U_3(X) = 8X^3 - 4X$.

Le but de cet appendice est de montrer les relations :

$$\boxed{\tau_n := T_n(-2) = \frac{(-1)^n}{2}\left((2+\sqrt{3})^n + (2-\sqrt{3})^n\right)}$$

$$\boxed{\mu_n := U_n(-2) = \frac{(-1)^n}{2\sqrt{3}}\left((2+\sqrt{3})^{n+1} - (2-\sqrt{3})^{n+1}\right)}$$

Proposition E.5 (P1 : Parité)**.** *Soit $n \in \mathbb{N}$. Alors*
(1) $T_n(-X) = (-1)^n T_n(X)$
(2) $U_n(-X) = (-1)^n U_n(X)$.

Démonstration. Soit $\theta \in \mathbb{R}$. Alors

$$T_n(-\cos\theta) = T_n(\cos(\theta+\pi)) = \cos(n\theta + n\pi) = (-1)^n \cos(n\theta) = (-1)^n T_n(\cos\theta)$$

et

$$\begin{aligned} \sin(\theta) \times U_n(-\cos(\theta)) &= -\sin(\theta+\pi) \times U_n(\cos(\theta+\pi)) \\ &= -\sin((n+1)\theta + (n+1)\pi) \\ &= (-1)^n \sin((n+1)\theta) = \sin(\theta) \times (-1)^n U_n(\cos(\theta)). \end{aligned}$$

Les polynômes sont égaux sur l'intervalle $[-1,1]$ donc sont égaux. □

Proposition E.6 (P2 : Relation entre T_{n+1} et U_n)**.** *Soit $n \in \mathbb{N}$. Alors*

$$T'_{n+1} = (n+1)U_n.$$

Démonstration. En dérivant la relation $T_{n+1}(\cos\theta) = \cos((n+1)\theta)$, on obtient

$$-\sin(\theta) T'_{n+1}(\cos\theta) = -(n+1)\sin((n+1)\theta)$$

soit

$$\sin(\theta)\left(\frac{T_{n+1}}{n+1}\right)'(\cos\theta) = \sin((n+1)\theta).$$

Par unicité de U_n, on a que $T'_{n+1} = (n+1)U_n$. □

Proposition E.7 (P3 : Relation de récurrence entre les T_n)**.**

$$\forall n \in \mathbb{N}, \quad T_{n+2}(X) - 2XT_{n+1}(X) + T_n(X) = 0.$$

Démonstration. Pout tout $\theta \in \mathbb{R}$ et $n \in \mathbb{N}$, on a

$$\cos(n\theta) + \cos((n+2)\theta) = 2\cos(\theta)\cos((n+1)\theta)$$

d'où

$$\forall \theta \in \mathbb{R}, \quad T_n(\cos\theta) + T_{n+2}(\cos\theta) = 2\cos(\theta) T_{n+1}(\cos\theta).$$

Les polynômes sont égaux sur l'intervalle $[-1,1]$ donc sont égaux. □

Proposition E.8 (P4 : Fonctions hyperboliques)**.** *Soit $n \in \mathbb{N}$. Alors*
(1) Pour tout $\theta \in \mathbb{R}$, $T_n(ch(\theta)) = ch(n\theta)$
(2) Pour tout $\theta \in \mathbb{R}$, $sh(\theta)U_n(ch(\theta)) = sh((n+1)\theta)$

Démonstration. (1) Par récurrence sur n. La relation est vraie pour $n = 0$ et $n = 1$. Supposons que la propriété est vraie aux rangs n et $n+1$ et montrons la au rang $n+2$. En utilisant la proposition P3 et l'hypothèse de récurrence, on obtient :

$$\begin{aligned} T_{n+2}(ch(\theta)) &= 2ch(\theta)T_{n+1}(ch(\theta)) - T_n(ch(\theta)) \\ &= 2ch(\theta)ch((n+1)\theta) - ch(n\theta) \\ &= ch((n+2)\theta) - ch(n\theta) + ch(n\theta) \\ &= ch((n+2)\theta). \end{aligned}$$

ce qui achève la récurrence.
(2) En dérivant l'égalité obtenue en (1), on obtient

$$sh(\theta)T'_{n+1}(ch(\theta)) = (n+1)sh((n+1)\theta).$$

D'après la propriété P2, on a

$$\left(\frac{T'_{n+1}}{n+1}\right)(ch(\theta)) = U_n(ch(\theta))$$

d'où

$$sh(\theta)U_n(ch(\theta)) = sh((n+1)\theta).$$

□

Proposition E.9 (P5 : Expression des T_n et U_n)**.** *Pour tout $n \in \mathbb{N}$ et tout $x > 1$, on a les relations :*

$$\begin{aligned} T_n(x) &= \frac{1}{2}\left((x+\sqrt{x^2-1})^n + (x-\sqrt{x^2-1})^n\right) \\ U_n(x) &= \frac{1}{2\sqrt{x^2-1}}\left((x+\sqrt{x^2-1})^{n+1} - (x-\sqrt{x^2-1})^{n+1}\right). \end{aligned}$$

Démonstration. (1) Soit $x > 1$. Notons $\theta = argch(x) = \ln(x+\sqrt{x^2-1})$. D'aprés la propriété P4, on obtient

$$\begin{aligned} T_n(x) &= T_n(ch(\theta)) \\ &= ch(n\theta) \\ &= \frac{1}{2}\left(e^{n\ln(x+\sqrt{x^2-1})} + e^{-n\ln(x+\sqrt{x^2-1})}\right) \\ &= \frac{1}{2}\left((x+\sqrt{x^2-1})^n + (x+\sqrt{x^2-1})^{-n}\right) \\ &= \frac{1}{2}\left((x+\sqrt{x^2-1})^n + (x-\sqrt{x^2-1})^n\right) \end{aligned}$$

en utilisant le fait que $(x+\sqrt{x^2-1})(x-\sqrt{x^2-1}) = 1$.

(2) En utilisant la propriété P2, on obtient par dérivation de la relation obtenue en (1) :

$$\begin{aligned} U_n(x) &= \frac{T'_{n+1}}{n+1}(x) \\ &= \frac{n+1}{2(n+1)}\left((1+\frac{x}{\sqrt{x^2-1}})(x+\sqrt{x^2-1})^n + (1-\frac{x}{\sqrt{x^2-1}})(x-\sqrt{x^2-1})^n\right) \\ &= \frac{1}{2\sqrt{x^2-1}}\left((x+\sqrt{x^2-1})^{n+1} - (x-\sqrt{x^2-1})^{n+1}\right). \end{aligned}$$

□

Finalement, d'après les propositions P1 et P5, on trouve

$$\begin{aligned} T_n(-2) &= (-1)^n T_n(2) = \frac{(-1)^n}{2}\left((2+\sqrt{3})^n + (2-\sqrt{3})^n\right) \\ U_n(-2) &= (-1)^n U_n(2) = \frac{(-1)^n}{2\sqrt{3}}\left((2+\sqrt{3})^{n+1} - (2-\sqrt{3})^{n+1}\right). \end{aligned}$$

F Equations équivalentes pour le schéma Lagrange+Projection

F.1 Calculs en advection constante

Calcul du premier terme de l'équation équivalente

Développement limité de la solution exacte.

Soit ρ la solution exacte de l'équation

$$\partial_t \rho + u_0 \partial_x \rho = 0. \tag{F.1}$$

Nous allons exprimer $\rho(x, t^{n+1})$ en fonction de ρ et de ses gradients évalués au temps t^n. En intégrant (F.1) entre t^n et t^{n+1}, nous avons

$$\rho(x, t^{n+1}) - \rho(x, t^n) = -u_0 \int_{t^n}^{t^{n+1}} \partial_x \rho(x, t) dt.$$

En faisant un développement limité du terme de droite autour de t^n puis en remplaçant les dérivées temporelles par des dérivées spatiales grâce à l'équation (F.1), nous obtenons :

$$\begin{aligned} \rho(x, t^{n+1}) - \rho(x, t^n) &= -u_0 \Delta t \partial_x \rho(x, t^n) + u_0^2 \frac{\Delta t^2}{2} \partial_x^2 \rho(x, t^n) \\ &\quad - u_0^3 \frac{\Delta t^3}{6} \partial_x^3 \rho(x, t^n) + u_0^4 \frac{\Delta t^4}{24} \partial_x^4 \rho(x, t^n) + o(\Delta t^4). \end{aligned}$$

Développement limité de la solution obtenue par le schéma.

Par définition,

$$r_i^n = \frac{\rho_i^n - \rho_{i-1}^n}{\rho_{i+1}^n - \rho_i^n}.$$

En notation continue, nous avons

$$\begin{aligned} \rho(x, t^n) - \rho(x - \Delta x, t^n) &= \Delta x \partial_x \rho(x, t^n) - \frac{\Delta x^2}{2} \partial_x^2 \rho(x, t^n) + \frac{\Delta x^3}{6} \partial_x^3 \rho(x, t^n) + o(\Delta x^3) \\ \rho(x + \Delta x, t^n) - \rho(x, t^n) &= \Delta x \partial_x \rho(x, t^n) + \frac{\Delta x^2}{2} \partial_x^2 \rho(x, t^n) + \frac{\Delta x^3}{6} \partial_x^3 \rho(x, t^n) + o(\Delta x^3) \end{aligned}$$

d'où

$$r(x,t^n) = \frac{\Delta x \partial_x \rho(x,t^n) - \frac{\Delta x^2}{2}\partial_x^2\rho(x,t^n) + \frac{\Delta x^3}{6}\partial_x^3\rho(x,t^n) + o(\Delta x^3)}{\Delta x \partial_x \rho(x,t^n) + \frac{\Delta x^2}{2}\partial_x^2\rho(x,t^n) + \frac{\Delta x^3}{6}\partial_x^3\rho(x,t^n) + o(\Delta x^3)}.$$

Dans le cas où $\partial_x\rho(x,t^n) \neq 0$, nous posons

$$F(x,t^n) = \frac{\partial_x^2\rho(x,t^n)}{\partial_x\rho(x,t^n)}, \qquad G(x,t^n) = \frac{\partial_x^3\rho(x,t^n)}{\partial_x\rho(x,t^n)}$$

d'où :

$$\begin{aligned} r(x,t^n) &= \frac{1 - \frac{\Delta x}{2}F(x,t^n) + \frac{\Delta x^2}{6}G(x,t^n) + o(\Delta x^2)}{1 + \frac{\Delta x}{2}F(x,t^n) + \frac{\Delta x^2}{6}G(x,t^n) + o(\Delta x^2)} \\ &= \left(1 - \frac{\Delta x}{2}F(x,t^n) + \frac{\Delta x^2}{6}G(x,t^n) + o(\Delta x^2)\right) \times \\ &\quad \left(1 - \frac{\Delta x}{2}F(x,t^n) - \frac{\Delta x^2}{6}G(x,t^n) + \frac{\Delta x^2}{4}F^2(x,t^n) + o(\Delta x^2)\right) \end{aligned}$$

donc

$$\boxed{r(x,t^n) = 1 - \Delta x F(x,t^n) + \tfrac{\Delta x^2}{2}F^2(x,t^n) + o(\Delta x^2).}$$

Connaissant le développement limité de r, nous pouvons en déduire ceux de Ψ^+ et Ψ^-.

$\star$ Si $r(x,t^n) \leq 0$ alors $\Psi^+(x,t^n) = \Psi^-(x,t^n) = 0$.

$\star$ Si $r(x,t^n) > 0$ alors

$$\begin{aligned} \Psi^+(x,t^n) &= \frac{2 \cdot r(x,t^n)}{1 + r(x,t^n)} \\ &= \frac{2 \cdot (1 - \Delta x F(x,t^n) + \frac{\Delta x^2}{2}F^2(x,t^n) + o(\Delta x^2))}{2 - \Delta x F(x,t^n) + \frac{\Delta x^2}{2}F^2(x,t^n) + o(\Delta x^2)} \\ &= \frac{1 - \Delta x F(x,t^n) + \frac{\Delta x^2}{2}F^2(x,t^n) + o(\Delta x^2)}{1 - \frac{\Delta x}{2}F(x,t^n) + \frac{\Delta x^2}{4}F^2(x,t^n) + o(\Delta x^2)} \\ &= \left(1 - \Delta x F(x,t^n) + \frac{\Delta x^2}{2}F^2(x,t^n) + o(\Delta x^2)\right) \times \\ &\quad \left(1 + \frac{\Delta x}{2}F(x,t^n) - \frac{\Delta x^2}{4}F^2(x,t^n) + \frac{\Delta x^2}{4}F^2(x,t^n) + o(\Delta x^2)\right) \end{aligned}$$

donc

$$\boxed{\Psi^+(x,t^n) = 1 - \tfrac{\Delta x}{2}F(x,t^n) + o(\Delta x^2)}$$

et

$$\begin{aligned}\Psi^-(x,t^n) &= \frac{2\cdot\frac{1}{r(x,t^n)}}{1+\frac{1}{r(x,t^n)}}\\ &= \frac{2}{r(x,t^n)+1}\\ &= \frac{2}{2-\Delta x F(x,t^n)+\frac{\Delta x^2}{2}F^2(x,t^n)+o(\Delta x^2)}\\ &= 1+\frac{\Delta x}{2}F(x,t^n)-\frac{\Delta x^2}{4}F^2(x,t^n)+\frac{\Delta x^2}{4}F^2(x,t^n)+o(\Delta x^2)\end{aligned}$$

donc

$$\boxed{\Psi^-(x,t^n)=1+\tfrac{\Delta x}{2}F(x,t^n)+o(\Delta x^2).}$$

De même, nous obtenons les résultats suivants :

$$\boxed{r(x-\Delta x,t^n)=1-\Delta x F(x,t^n)+\tfrac{\Delta x^2}{2}[2G(x,t^n)-F^2(x,t^n)]+o(\Delta x^2)}$$

$$\boxed{\Psi^+(x-\Delta x,t^n)=1-\tfrac{\Delta x}{2}F(x,t^n)+\tfrac{\Delta x^2}{2}[G(x,t^n)-F^2(x,t^n)]+o(\Delta x^2)}$$

$$\boxed{\Psi^-(x-\Delta x,t^n)=1+\tfrac{\Delta x}{2}F(x,t^n)-\tfrac{\Delta x^2}{2}[G(x,t^n)-F^2(x,t^n)]+o(\Delta x^2).}$$

Pour simplifier les formules, notons $C=(1-\eta)/4$. Le schéma en advection constante se réécrit sous la forme :

$$\frac{\rho_i^{n+1}-\rho_i^n}{\Delta t}+$$

$$\frac{u_0}{\Delta x}\left(C\Psi_i^+\rho_{i+1}^n+[1+C(-\Psi_i^++\Psi_i^--\Psi_{i-1}^+)]\rho_i^n+[-1+C(-\Psi_i^-+\Psi_{i-1}^+-\Psi_{i-1}^-)]\rho_{i-1}^n+C\Psi_{i-1}^-\rho_{i-2}^n\right)=0$$

Les développements limités obtenus précédemment pour $\Psi_i^+,\Psi_i^-,\Psi_{i-1}^+,\Psi_{i-1}^-$ nous permettent d'exprimer l'erreur de troncature en fonction de ρ et de ses gradients évalués au temps t^n dans les différents cas, en supposant que la solution exacte et la solution donnée par le schéma coïncident au temps t^n et en faisant un développement limité de la différence au temps t^{n+1}.

Premier cas : $r_{i-1}\leq 0$ et $r_i\leq 0$

Dans ce cas, $\Psi_{i-1}^-=\Psi_{i-1}^+=\Psi_i^-=\Psi_i^+=0$.

Ainsi,

$$\rho_i^{n+1}-\rho_i^n=-\eta\left(\Delta x\partial_x\rho(x_i,t^n)-\frac{\Delta x^2}{2}\partial_x^2\rho(x_i,t^n)\right)+o(\Delta x^2).$$

Nous allons utiliser ce résultat ainsi que celui de la première partie sur la solution exacte pour calculer l'erreur de troncature du schéma :

$$\varepsilon(x_i, t^{n+1}) = \frac{1}{\Delta t}\Big(\rho_i^{n+1} - \rho(x_i, t^{n+1})\Big).$$

Le premier terme dans le développement limité de cette erreur sera le premier terme dans l'équation équivalente. Nous obtienons alors :

$$\varepsilon = \frac{\Delta x}{2} u_0(1-\eta)\partial_x^2\rho + o(\Delta x).$$

Il s'agit de la diffusion numérique du schéma Upwind.

Second cas : $r_{i-1} \leq 0$ et $r_i > 0$

Dans ce cas, nous avons $\Psi_{i-1}^- = \Psi_{i-1}^+ = 0$ et

$$\Psi^-(x_i, t^n) = 1 + \frac{\Delta x}{2}F(x_i, t^n) + o(\Delta x^2), \qquad \Psi^+(x_i, t^n) = 1 - \frac{\Delta x}{2}F(x_i, t^n) + o(\Delta x^2).$$

Ainsi,

$$\rho_i^{n+1} - \rho_i^n = -\eta\left(\Delta x \frac{3-\eta}{2}\partial_x\rho(x_i, t^n)\right) + o(\Delta x).$$

Nous obtenons alors :

$$\varepsilon = u_0\frac{\eta - 1}{2}\partial_x\rho + o(1).$$

Troisième cas : $r_{i-1} > 0$ et $r_i \leq 0$

Dans ce cas, nous avons $\Psi_i^- = \Psi_i^+ = 0$ et

$$\begin{aligned}
\Psi^-(x_{i-1}, t^n) &= 1 + \frac{\Delta x}{2}F(x_i, t^n) - \frac{\Delta x^2}{2}[G(x_i, t^n) - F^2(x_i, t^n)] + o(\Delta x^2) \\
\Psi^+(x_{i-1}, t^n) &= 1 - \frac{\Delta x}{2}F(x_i, t^n) + \frac{\Delta x^2}{2}[G(x_i, t^n) - F^2(x_i, t^n)] + o(\Delta x^2).
\end{aligned}$$

Ainsi,

$$\tilde\rho(x_i, t^{n+1}) - \tilde\rho(x_i, t^n) = -\eta\left(\Delta x \frac{1+\eta}{2}\partial_x\tilde\rho(x_i, t^n)\right) + o(\Delta x).$$

Nous obtenons alors :

$$\varepsilon = u_0\frac{1-\eta}{2}\partial_x\rho + o(1).$$

Quatrième cas : $r_{i-1} > 0$ et $r_i > 0$

Dans ce cas, nous utilisons :

$$\begin{aligned}
\Psi^-(x_i,t^n) &= 1+\frac{\Delta x}{2}F(x_i,t^n)+o(\Delta x^2)\\
\Psi^+(x_i,t^n) &= 1-\frac{\Delta x}{2}F(x_i,t^n)+o(\Delta x^2)\\
\Psi^-(x_{i-1},t^n) &= 1+\frac{\Delta x}{2}F(x_i,t^n)-\frac{\Delta x^2}{2}[G(x_i,t^n)-F^2(x_i,t^n)]+o(\Delta x^2)\\
\Psi^+(x_{i-1},t^n) &= 1-\frac{\Delta x}{2}F(x_i,t^n)+\frac{\Delta x^2}{2}[G(x_i,t^n)-F^2(x_i,t^n)]+o(\Delta x^2).
\end{aligned}$$

Ainsi,

$$\rho_i^{n+1}-\rho_i^n=-\eta\left(\Delta x\partial_x\rho(x_i,t^n)-\Delta x^2\frac{\eta}{2}\partial_x^2\rho(x_i,t^n)+\Delta x^3\frac{3\eta-1}{12}\partial_x^3\rho(x_i,t^n)\right)+o(\Delta x^3).$$

Nous obtenons alors :

$$\varepsilon=u_0\frac{\Delta x^2}{12}(2\eta^2-3\eta+1)\partial_x^3\rho+o(\Delta x^2).$$

Calcul du second terme de l'équation équivalente

Comme nous voulons obtenir le premier terme de diffusion, il nous faut encore calculer le second terme de l'équation équivalente dans les cas

1. $r_{i-1}\leq 0, r_i>0$
2. $r_{i-1}>0, r_i\leq 0$
3. $r_{i-1}>0, r_i>0$

Cas $r_{i-1}\leq 0,\ r_i>0$

Ce second terme sera la somme du terme en Δx de l'erreur de troncature et du terme d'ordre 1 provenant du terme d'ordre 0. Le terme en Δx de l'erreur de troncature vaut (que l'on calcule comme dans la partie précédente) :

$$\varepsilon-u_0\frac{\eta-1}{2}\partial_x\rho=u_0\Delta x\frac{1-\eta}{2}\partial_x^2\rho+o(\Delta x).$$

Pour calculer le terme provenant du terme d'ordre 0, nous calculons les développements limités des solutions des équations :

$$\begin{aligned}
\partial_t\rho+u_0\partial_x\rho &= 0\\
\partial_t\tilde{\rho}+u_0\partial_x\tilde{\rho} &= u_0\frac{\eta-1}{2}\partial_x\tilde{\rho}
\end{aligned}$$

puis nous calculons le terme d'ordre 1 de la différence :

$$\frac{1}{\Delta t}\left(\tilde{\rho}(x_i,t^{n+1})-\rho(x_i,t^{n+1})\right).$$

Au final, nous obtienons le terme suivant de l'équation équivalente :

$$\Pi = u_0 \frac{\eta - 1}{2} \partial_x \rho + u_0 \Delta x \frac{(4-\eta)(\eta-1)^2}{8} \partial_x^2 \rho.$$

Cas $r_{i-1} > 0,\ r_i \leq 0$

Ce second terme sera la somme du terme en Δx de l'erreur de troncature et du terme d'ordre 1 provenant du terme d'ordre 0. Le terme en Δx de l'erreur de troncature vaut 0 :

$$\varepsilon - u_0 \frac{1-\eta}{2} \partial_x \rho = o(\Delta x).$$

Pour calculer le terme provenant du terme d'ordre 0, on calcule les développements limités des solutions des équations :

$$\begin{aligned} \partial_t \rho + u_0 \partial_x \rho &= 0 \\ \partial_t \tilde{\rho} + u_0 \partial_x \tilde{\rho} &= u_0 \frac{1-\eta}{2} \partial_x \tilde{\rho} \end{aligned}$$

puis nous calculons le terme d'ordre 1 de la différence :

$$\frac{1}{\Delta t} \Big(\tilde{\rho}(x_i, t^{n+1}) - \rho(x_i, t^{n+1}) \Big).$$

Au final, nous obtienons les termes suivants dans l'équation équivalente :

$$\Pi = u_0 \frac{1-\eta}{2} \partial_x \rho + u_0 \Delta x \frac{\eta(1-\eta)(3+\eta)}{8} \partial_x^2 \rho.$$

Cas $r_{i-1} > 0,\ r_i > 0$

Ce second terme sera la somme du terme en Δx^3 de l'erreur de troncature et du terme d'ordre 3 provenant du terme d'ordre 2. Le terme en Δx^3 de l'erreur de troncature vaut :

$$\varepsilon - u_0 \frac{\Delta x^2}{12} (2\eta^2 - 3\eta + 1) \partial_x^3 \rho = u_0 \frac{\Delta x^3}{24} (\eta^3 - 4\eta + 3) \partial_x^4 + o(\Delta x^3).$$

Pour calculer le terme provenant du terme d'ordre 2, nous calculons les développements limités des solutions des équations :

$$\begin{aligned} \partial_t \rho + u_0 \partial_x \rho &= 0 \\ \partial_t \tilde{\rho} + u_0 \partial_x \tilde{\rho} &= u_0 \frac{\Delta x^2}{12} (2\eta^2 - 3\eta + 1) \partial_x^3 \tilde{\rho} \end{aligned}$$

puis nous calculons le terme d'ordre 3 de la différence :

$$\frac{1}{\Delta t} \Big(\tilde{\rho}(x_i, t^{n+1}) - \rho(x_i, t^{n+1}) \Big).$$

Au final, nous obtienons les termes suivants dans l'équation équivalente :

$$\Pi = u_0 \frac{\Delta x^2}{12} (2\eta^2 - 3\eta + 1) \partial_x^3 \rho + u_0 \frac{\Delta x^3}{8} (\eta - 1)(\eta^2 - \eta + 1) \partial_x^4 \rho.$$

F.2 Calculs en advection non constante

Calcul du premier terme de l'équation équivalente

Développement limité de la solution exacte

Soit ρ la solution exacte de l'équation

$$\partial_t \rho + \partial_x(u\rho) = 0. \tag{F.2}$$

En intégrant (F.2) entre t^n et t^{n+1}, on a

$$\rho(x,t^{n+1}) - \rho(x,t^n) = -\int_{t^n}^{t^{n+1}} \partial_x(u\rho)(x,t)dt$$

Le programme Maple (Appendice F.3) nous permet d'obtenir le développement limité de cette dernière intégrale et nous avons finalement :

$$\begin{aligned}\rho(x,t^{n+1}) - \rho(x,t^n) &= -\Delta t\Big((\partial_x u)(x)\rho(x,t^n) + u(x)(\partial_x\rho)(x,t^n)\Big) \\ &\quad + \frac{\Delta t^2}{2}(...) - \frac{\Delta t^3}{6}(...) + \frac{\Delta t^4}{24}(...) + o(\Delta t^4)\end{aligned}$$

où les coefficients (...) sont donnés par les dernières commandes du programme Maple.

Développement limité de la solution obtenue par le schéma

Le schéma s'écrit

$$\rho_i^{n+1} - \rho_i^n = -\frac{dm_{i+1/2} - dm_{i-1/2}}{\Delta x}.$$

Le programme Maple (Appendice F.3) nous permet d'obtenir le développement limité du terme de droite. Pour cela, nous choisissons dans le programme si nous considérons un limiteur sur la cellule $i-1$ et/ou sur la cellule i :

```
[Exemple avec limiteur sur la cellule i-1 et sur la cellule i]
> dpli := (Psi1moinsTAYLOR*Aim1TAYLOR+Psi1plusTAYLOR*AiTAYLOR)*(1/2):
  dpliTAYLOR := taylor(dpli, x, n): # cas avec limiteur sur la cellule i
> # dpliTAYLOR := 0: # cas sans limiteur sur la cellule i
> dplim1 := (Psi0moinsTAYLOR*Aim2TAYLOR+Psi0plusTAYLOR*Aim1TAYLOR)*(1/2):
  dplim1TAYLOR := taylor(dplim1, x, n): # cas avec limiteur sur la cellule i-1
> # dplim1TAYLOR := 0: # cas sans limiteur sur la cellule i-1
```

et finalement le programme nous donne le coefficient devant $\Delta x^i \Delta t^j$ dans le développement limité du terme de droite.

Par exemple dans le cas avec limiteur sur la cellule $i-1$ et sur la cellule i, nous obtenons :

$$\rho_i^{n+1} - \rho_i^n = -\Delta t\Big((\partial_x u)(x_i)\rho(x_i,t^n) + u(x_i)(\partial_x\rho)(x_i,t^n)\Big) + o(\Delta t) + o(\Delta x).$$

Le premier terme dans le développement limité de l'erreur de troncature sera le premier terme de l'équation équivalente. Nous trouvons alors les termes suivants :

Cas où $r_{i-1} \leq 0$ et $r_i \leq 0$ (pas de limiteur sur les cellules $i-1$ et i)

$$\varepsilon = \frac{\Delta x}{2}\left((\partial_x u)(\partial_x \rho) + u\partial_{xx}\rho\right) + \frac{\Delta t}{2}\left(\rho(\partial_x u)^2 - u(\partial_x u)(\partial_x \rho) + \rho u \partial_{xx} u - u^2 \partial_{xx}\rho\right) + o(\Delta t) + o(\Delta x).$$

Cas où $r_{i-1} \leq 0$ et $r_i > 0$

$$\varepsilon = u\frac{\frac{\Delta t}{\Delta x}u - 1}{2}\partial_x \rho + o(1).$$

Cas où $r_{i-1} > 0$ et $r_i \leq 0$

$$\varepsilon = u\frac{1 - \frac{\Delta t}{\Delta x}u}{2}\partial_x \rho + o(1).$$

Cas où $r_{i-1} > 0$ et $r_i > 0$

$$\varepsilon = \frac{\Delta t}{2}\left(\rho(\partial_x u)^2 + u(\partial_x \rho)(\partial_x u) + u\rho(\partial_{xx} u)\right) + o(\Delta t) + o(\Delta x).$$

Calcul du second terme de l'équation équivalente

D'après les résultats précédents, il nous faut encore calculer le second terme de l'équation équivalente dans les cas $r_{i-1} \leq 0, r_i > 0$ et $r_{i-1} > 0, r_i \leq 0$. Ce second terme sera la somme des termes en Δt et en Δx de l'erreur de troncature et du terme d'ordre 1 provenant du terme d'ordre 0. Les termes en Δt et en Δx de l'erreur de troncature valent :

Cas où $r_{i-1} \leq 0$ et $r_i > 0$

$$\begin{aligned}\varepsilon - u\frac{\frac{\Delta t}{\Delta x}u - 1}{2}\partial_x \rho &= \Delta t\left(\frac{1}{2}(\partial_x u)^2 \rho + \frac{1}{2}u(\partial_x u)(\partial_x \rho) + u\rho(\partial_{xx} u) - \frac{1}{2}u^2(\partial_{xx}\rho)\right) \\ &\quad + \Delta x\left(\frac{1}{2}u(\partial_{xx}\rho) + \frac{1}{4}(\partial_x \rho)(\partial_x u)\right) + o(\Delta x) + o(\Delta t).\end{aligned}$$

Cas où $r_{i-1} > 0$ et $r_i \leq 0$

$$\varepsilon - u\frac{1 - \frac{\Delta t}{\Delta x}u}{2}\partial_x \rho = \Delta t\left(\frac{1}{2}(\partial_x u)^2 \rho - \frac{1}{2}u(\partial_x u)(\partial_x \rho)\right) + \Delta x\left(\frac{1}{4}(\partial_x \rho)(\partial_x u)\right) + o(\Delta x) + o(\Delta t).$$

Pour calculer le terme provenant du terme d'ordre 0, nous calculons les développements

limités des solutions des équations :

$$\begin{aligned}
\partial_t\rho + \partial_x(u\rho) &= 0 \\
\partial_t\tilde{\rho} + \partial_x(u\tilde{\rho}) &= u\frac{\frac{\Delta t}{\Delta x}u - 1}{2}\partial_x\tilde{\rho} \qquad [Cas\ 1 : r_{i-1} \leq 0, r_i > 0] \\
\partial_t\tilde{\rho} + \partial_x(u\tilde{\rho}) &= u\frac{1 - \frac{\Delta t}{\Delta x}u}{2}\partial_x\tilde{\rho} \qquad [Cas\ 2 : r_{i-1} > 0, r_i \leq 0]
\end{aligned}$$

puis nous calculons le terme d'ordre 1 de la différence :

$$\frac{1}{\Delta t}\left(\tilde{\rho}(x_i, t^{n+1}) - \rho(x_i, t^{n+1})\right).$$

Ces calculs sont faits par le programme Maple et nous obtenons les termes suivants dans l'équation équivalente :

Cas où $r_{i-1} \leq 0$ et $r_i > 0$

$$\begin{aligned}
\Pi &= u\frac{\frac{\Delta t}{\Delta x}u - 1}{2}\partial_x\rho + \Delta t\left(\frac{1}{2}(\partial_x u)^2\rho + \frac{5}{8}u(\partial_x u)(\partial_x\rho) + \frac{3}{4}u\rho(\partial_{xx}u) - \frac{9}{8}u^2(\partial_{xx}\rho) + \frac{13}{8}\frac{\Delta t}{\Delta x}u^2(\partial_x\rho)(\partial_x u)\right. \\
&\quad + \frac{3}{4}\frac{\Delta t}{\Delta x}u^3(\partial_{xx}\rho) + \frac{1}{4}\frac{\Delta t}{\Delta x}u^2(\partial_{xx}u)\rho - \frac{1}{4}u^3\left(\frac{\Delta t}{\Delta x}\right)^2 u^3(\partial_x\rho)(\partial_x u) - \frac{1}{8}\left(\frac{\Delta t}{\Delta x}\right)^2 u^4(\partial_{xx}\rho)\Big) \\
&\quad + \Delta x\left(\frac{1}{2}u(\partial_{xx}\rho) + \frac{1}{4}(\partial_x u)(\partial_x\rho)\right).
\end{aligned}$$

Cas où $r_{i-1} > 0$ et $r_i \leq 0$

$$\begin{aligned}
\Pi &= u\frac{1 - \frac{\Delta t}{\Delta x}u}{2}\partial_x\rho + \Delta t\left(\frac{1}{2}(\partial_x u)^2\rho + \frac{3}{8}u(\partial_x u)(\partial_x\rho) + \frac{1}{4}u\rho(\partial_{xx}u) + \frac{3}{8}u^2(\partial_{xx}\rho) - \frac{7}{8}\frac{\Delta t}{\Delta x}u^2(\partial_x\rho)(\partial_x u)\right. \\
&\quad - \frac{1}{4}\frac{\Delta t}{\Delta x}u^3(\partial_{xx}\rho) - \frac{1}{4}\frac{\Delta t}{\Delta x}u^2(\partial_{xx}u)\rho - \frac{1}{4}\left(\frac{\Delta t}{\Delta x}\right)^2 u^3(\partial_x\rho)(\partial_x u) - \frac{1}{8}\left(\frac{\Delta t}{\Delta x}\right)^2 u^4(\partial_{xx}\rho)\Big) \\
&\quad + \frac{\Delta x}{4}(\partial_x u)(\partial_x\rho).
\end{aligned}$$

F.3 Code Maple

Nous donnons ici le code Maple permettant d'obtenir les résultats de la partie précédente en advection non constante.

Calcul du premier terme de l'équation équivalente

Développement limité de la solution exacte

```
> restart:
> g := (x, t) -> u(x)*p(x, t): pt1 := -D[1](g):
> pt2 := (x, t) -> subs((D[2](p))(x, t) = pt1(x, t),
  (D[1, 2](p))(x, t) = (D[1](pt1))(x, t), (D[2](pt1))(x, t)):
> pt3 := (x, t) -> subs((D[2](p))(x, t) = pt1(x, t),
  (D[1, 2](p))(x, t) = (D[1](pt1))(x, t), (D[1, 1, 2](p))(x, t) = (D[1, 1](pt1))(x, t
  (D[2](pt2))(x, t)):
> pt4 := (x, t) -> subs((D[2](p))(x, t) = pt1(x, t),
  (D[1, 2](p))(x, t) = (D[1](pt1))(x, t), (D[1, 1, 2](p))(x, t) = (D[1, 1](pt1))(x, t
  (D[1, 1, 1, 2](p))(x, t) = (D[1, 1, 1](pt1))(x, t), (D[2](pt3))(x, t)):
> p0 := (x, t) -> p(x, a)+pt1(x, a)*(t-a)+(1/2)*pt2(x, a)*(t-a)^2+(1/6)*pt3(x, a)*(t-
  (1/24)*pt4(x, a)*(t-a)^4:
> p1 := (x, t) -> (D[1](p))(x, a)+(D[1](pt1))(x, a)*(t-a)+(1/2)*(D[1](pt2))(x, a)*(t-
  (1/6)*(D[1](pt3))(x, a)*(t-a)^3+(1/24)*(D[1](pt4))(x, a)*(t-a)^4:
> int(p0(x, t), t = a .. a+Delta):
> Int1 := %:
> int(p1(x, t), t = a .. a+Delta):
> Int2 := %:
> final := -(D[1](u))(x)*Int1-u(x)*Int2:
> expand(coeff(final, Delta, 0));
> expand(coeff(final, Delta, 1));
> expand(coeff(final, Delta, 2));
> expand(coeff(final, Delta, 3));
> expand(coeff(final, Delta, 4));
```

Développement limité de la solution obtenue par le schéma

```
> restart:
> n := 6:
> denomAim2 := x+(1/2)*t*(taylor(u(y-(1/2)*x), x, n)-taylor(u(y-5*x*(1/2)), x, n)):
  denomAim2TAYLOR := taylor(denomAim2, x, n):
> denomAim1 := x+(1/2)*t*(taylor(u(y+(1/2)*x), x, n)-taylor(u(y-3*x*(1/2)), x, n)):
  denomAim1TAYLOR := taylor(denomAim1, x, n):
> denomAi := x+(1/2)*t*(taylor(u(y+3*x*(1/2)), x, n)-taylor(u(y-(1/2)*x), x, n)):
  denomAiTAYLOR := taylor(denomAi, x, n):
> volim2 := x+t*(taylor(u(y-3*x*(1/2)), x, n)-taylor(u(y-5*x*(1/2)), x, n)):
  volim2TAYLOR := taylor(volim2, x, n):
> volim1 := x+t*(taylor(u(y-(1/2)*x), x, n)-taylor(u(y-3*x*(1/2)), x, n)):
```

```
  volim1TAYLOR := taylor(volim1, x, n):
> voli := x+t*(taylor(u(y+(1/2)*x), x, n)-taylor(u(y-(1/2)*x), x, n)):
  voliTAYLOR := taylor(voli, x, n):
> volip1 := x+t*(taylor(u(y+3*x*(1/2)), x, n)-taylor(u(y+(1/2)*x), x, n)):
  volip1TAYLOR := taylor(volip1, x, n):
> plim2 := taylor(p(y-2*x), x, n)*x/volim2TAYLOR:
  plim2TAYLOR := taylor(plim2, x, n):
> plim1 := taylor(p(y-x), x, n)*x/volim1TAYLOR:
  plim1TAYLOR := taylor(plim1, x, n):
> pli := p(y)*x/voliTAYLOR:
  pliTAYLOR := taylor(pli, x, n):
> plip1 := taylor(p(y+x), x, n)*x/volip1TAYLOR:
  plip1TAYLOR := taylor(plip1, x, n):
> Aim2 := (plim1TAYLOR-plim2TAYLOR)/denomAim2TAYLOR:
  Aim2TAYLOR := taylor(Aim2, x, n):
> Aim1 := (pliTAYLOR-plim1TAYLOR)/denomAim1TAYLOR:
  Aim1TAYLOR := taylor(Aim1, x, n):
> Ai := (plip1TAYLOR-pliTAYLOR)/denomAiTAYLOR:
  AiTAYLOR := taylor(Ai, x, n):
> ri := Aim1TAYLOR/AiTAYLOR:
  riTAYLOR := taylor(ri, x, n):
> rim1 := Aim2TAYLOR/Aim1TAYLOR:
  rim1TAYLOR := taylor(rim1, x, n):
> Psi0plus := 2*rim1TAYLOR/(1+rim1TAYLOR):
  Psi0plusTAYLOR := taylor(Psi0plus, x, n):
> Psi0moins := 2/(rim1TAYLOR*(1+1/rim1TAYLOR)):
  Psi0moinsTAYLOR := taylor(Psi0moins, x, n):
> Psi1plus := 2*riTAYLOR/(1+riTAYLOR):
  Psi1plusTAYLOR := taylor(Psi1plus, x, n):
> Psi1moins := 2/(riTAYLOR*(1+1/riTAYLOR)):
  Psi1moinsTAYLOR := taylor(Psi1moins, x, n):
> dpli := (Psi1moinsTAYLOR*Aim1TAYLOR+Psi1plusTAYLOR*AiTAYLOR)*(1/2):
  dpliTAYLOR := taylor(dpli, x, n): # cas avec limiteur sur la cellule i
> # dpliTAYLOR := 0: # cas sans limiteur sur la cellule i
> dplim1 := (Psi0moinsTAYLOR*Aim2TAYLOR+Psi0plusTAYLOR*Aim1TAYLOR)*(1/2):
  dplim1TAYLOR := taylor(dplim1, x, n): # cas avec limiteur sur la cellule i-1
> # dplim1TAYLOR := 0: # cas sans limiteur sur la cellule i-1
> dmi := t*taylor(u(y+(1/2)*x), x, n)*(pliTAYLOR+(1/2)*dpliTAYLOR*
  (voliTAYLOR-t*taylor(u(y+(1/2)*x), x, n))):
  dmiTAYLOR := taylor(dmi, x, n):
> dmim1 := t*taylor(u(y-(1/2)*x), x, n)*(plim1TAYLOR+(1/2)*dplim1TAYLOR*
  (volim1TAYLOR-t*taylor(u(y-(1/2)*x), x, n))):
  dmim1TAYLOR := taylor(dmim1, x, n):
> final := -(dmiTAYLOR-dmim1TAYLOR)/x:finalTAYLOR := series(final, x):
> i:=1:
> j:=1:
```

```
> simplify(coeff(taylor(simplify(coeff(finalTAYLOR, x, i)), t), t, j));
```

Calcul du second terme de l'équation équivalente

```
> restart:
> n := 2:

> g := (x, t) -> u(x)*p(x, t): pt1 := D[1](g):
> part11 := taylor(p(x, t), t = a, n):
> part12 := subs((D[2](p))(x, a) = -pt1(x, a), part11):
> part13 := taylor((D[1](p))(x, t), t = a, n):
> part14 := subs((D[1, 2](p))(x, a) = -(D[1](pt1))(x, a), part13):
> part15 := subs(p(x, t) = part12, (D[1](p))(x, t) = part14, pt1(x, t)):
> p1 := (x, t) -> #mettre ici le résultat du calcul précédent
> Int1 := simplify(int(p1(x, t), t = a .. a+Delta)):

> h := (x, t) -> u(x)*p(x, t) end proc; pt2 := D[1](h):
> hbis := (x, t) -> u(x)*(D[1](p))(x, t) end proc; pt2bis := D[1](hbis):
> hbisbis := (x, t) -> u(x)^2*(D[1](p))(x, t) end proc; pt2bisbis := D[1](hbisbis):
> part21 := taylor(p(x, t), t = a, n):
> part22 := subs((D[2](p))(x, a) = -pt2(x, a)+
  (1/2)*u(x)^2*C*(D[1](p))(x, a)-(1/2)*u(x)*(D[1](p))(x, a), part21): # cas 1
> #part22 := subs((D[2](p))(x, a) = -pt2(x, a)-
  (1/2)*u(x)^2*C*(D[1](p))(x, a)+(1/2)*u(x)*(D[1](p))(x, a), part21): # cas 2
> part23 := taylor((D[1](p))(x, t), t = a, n):
> part24 := subs((D[1, 2](p))(x, a) = -(D[1](pt2))(x, a)+
  (1/2)*C*pt2bisbis(x, a)-(1/2)*pt2bis(x, a), part23):
> part25 := subs(p(x, t) = part22, (D[1](p))(x, t) = part24, pt2(x, t)):
> p2 := (x, t) -> #mettre ici le résultat du calcul précédent
> Int2 := simplify(int(p2(x, t), t = a .. a+Delta)):

> part33 := taylor((D[1](p))(x, t), t = a, n):
> part34 := subs((D[1, 2](p))(x, a) = -(D[1](pt2))(x, a)+
  (1/2)*C*pt2bisbis(x, a)-(1/2)*pt2bis(x, a), part33): # cas 1
> #part34 := subs((D[1, 2](p))(x, a) = -(D[1](pt2))(x, a)-
  (1/2)*C*pt2bisbis(x, a)+(1/2)*pt2bis(x, a), part33): # cas 2
> part35 := u(x)^2*part34:
> p3 := (x, t) -> #mettre ici le résultat du calcul précédent
> Int3 := simplify(int(p3(x, t), t = a .. a+Delta)):

> part35bis := u(x)*part34:
> p4 := (x, t) -> #mettre ici le résultat du calcul précédent
> Int4 := simplify(int(p4(x, t), t = a .. a+Delta)):

> final := expand((-Int1+Int2-(1/2)*C*Int3+(1/2)*Int4)/Delta): # cas 1
> #final := expand((-Int1+Int2+(1/2)*C*Int3-(1/2)*Int4)/Delta): # cas 2
```

```
> coeff(final, Delta, 1):
```

G Dérivation de la relation de dispersion

Afin de valider la partie linéaire des résultats numériques, nous calculons la relation de dispersion avec la gyromoyenne. Nous procédons aux développements suivants :

$$f = f_0 + \varepsilon f_1 + \mathcal{O}(\varepsilon^2), \quad \phi = \phi_0 + \varepsilon\phi_1 + \mathcal{O}(\varepsilon^2)$$

avec

$$f_0(r,v) = f_{eq}(r,v) = \frac{n_0(r)\exp\left(-\frac{v^2}{2T_i(r)}\right)}{(2\pi T_i(r))^{1/2}}, \quad \phi_0 = 0.$$

Ainsi, on obtient

$$\mathcal{J}_{\sqrt{2\mu}}(f) = \mathcal{J}_{\sqrt{2\mu}}(f_0 + \varepsilon f_1) + \mathcal{O}(\varepsilon^2),$$

et

$$\mathcal{J}_{\sqrt{2\mu}}(\phi) = \mathcal{J}_{\sqrt{2\mu}}(\bar{\phi_0} + \varepsilon\bar{\phi_1}) + \mathcal{O}(\varepsilon^2).$$

En substituant les relations ci-dessus dans (6.5.8), on obtient

$$\partial_t f_1 - \frac{\partial_\theta \mathcal{J}_{\sqrt{2\mu}}(\phi_1)}{r}\partial_r f_0 + v\partial_z f_1 - \partial_z \mathcal{J}_{\sqrt{2\mu}}(\phi_1)\partial_v f_0 = \mathcal{O}(\varepsilon).$$

De manière similaire, l'équation

$$-\left(\partial_r^2\phi + \left(\frac{1}{r} + \frac{\partial_r n_0(r)}{n_0(r)}\right)\partial_r\phi + \frac{1}{r^2}\partial_\theta^2\phi\right) + \frac{1}{T_e(r)}(\phi - \lambda\langle\phi\rangle) = \frac{1}{n_0(r)}\mathcal{J}_{\sqrt{2\mu}}\left(\int f - f_{eq}dv\right)$$

devient

$$-\left(\partial_r^2\phi_1 + \left(\frac{1}{r} + \frac{\partial_r n_0(r)}{n_0(r)}\right)\partial_r\phi_1 + \frac{1}{r^2}\partial_\theta^2\phi_1\right) + \frac{1}{T_e(r)}(\phi_1 - \lambda\langle\phi_1\rangle) = \frac{1}{n_0(r)}\mathcal{J}_{\sqrt{2\mu}}\left(\int f_1 dv\right) + \mathcal{O}(\varepsilon). \tag{G.1}$$

On suppose que les solutions sont sous la forme :

$$f_1 = f_{m,n,\omega}(r,v)e^{i(m\theta+kz-\omega t)}, \quad \phi_1 = \phi_{m,n,\omega}(r)e^{i(m\theta+kz-\omega t)}$$

$$\begin{aligned}\mathcal{J}_{\sqrt{2\mu}}(f_1) &= \hat{f}_{m,n,\omega}(r,v)e^{i(m\theta+kz-\omega t)},\\ \mathcal{J}_{\sqrt{2\mu}}(\phi_1) &= \hat{\phi}_{m,n,\omega}(r)e^{i(m\theta+kz-\omega t)}\end{aligned}$$

avec $k = \frac{2\pi n}{L}$. Ainsi, on obtient

$$(-\omega + kv)f_{m,n,\omega} = \left(\frac{m}{r}\partial_r f_0 + k\partial_v f_0\right)\hat{\phi}_{m,n,\omega} \tag{G.2}$$

et la relation (G.1) devient

$$-\left(\partial_r^2\phi_{m,n,\omega}+\left(\frac{1}{r}+\frac{\partial_r n_0(r)}{n_0(r)}\right)\partial_r\phi_{m,n,\omega}-\frac{m^2}{r^2}\phi_{m,n,\omega}\right)+\frac{1}{T_e(r)}(\phi_{m,n,\omega}-\lambda\delta_n^0\phi_{m,0,\omega})=\frac{1}{n_0(r)}\int\hat{f}_{m,n,\omega}dv.$$

Si nous supposons que $m\neq 0$ et $n\neq 0$, la dernière relation et l'équation (G.2) conduisent à :

$$-\left(\partial_r^2\phi_{m,n,\omega}+\left(\frac{1}{r}+\frac{\partial_r n_0(r)}{n_0(r)}\right)\partial_r\phi_{m,n,\omega}-\frac{m^2}{r^2}\phi_{m,n,\omega}\right)+\frac{1}{T_e(r)}\phi_{m,n,\omega}=\frac{1}{n_0(r)}\hat{\phi}_{m,n,\omega}\int\frac{\hat{f}_{m,n,\omega}}{f_{m,n,\omega}}\frac{\frac{m}{r}\partial_r f_0+k\partial_v f_0}{kv-\omega}dv$$

et donc

$$-\left(\frac{\partial_r^2\phi_{m,n,\omega}}{\phi_{m,n,\omega}}+\left(\frac{1}{r}+\frac{\partial_r n_0(r)}{n_0(r)}\right)\frac{\partial_r\phi_{m,n,\omega}}{\phi_{m,n,\omega}}-\frac{m^2}{r^2}\right)+\frac{1}{T_e(r)}=\frac{1}{n_0(r)}\frac{\hat{\phi}_{m,n,\omega}}{\phi_{m,n,\omega}}\int\frac{\hat{f}_{m,n,\omega}}{f_{m,n,\omega}}\frac{\frac{m}{r}\partial_r f_0+k\partial_v f_0}{kv-\omega}dv.$$

Nous faisons les approximations :

$$\frac{\hat{\Phi}_{m,n,\omega}}{\Phi_{m,n,\omega}}\approx J_0(\kappa\sqrt{2\mu}),\quad \frac{\hat{f}_{m,n,\omega}}{f_{m,n,\omega}}\approx J_0(\kappa\sqrt{2\mu})$$

où $\kappa\in\mathbb{R}^+$. Rigoureusement les approximations précédentes sont correctes lorsque f et Φ sont des fonctions de Fourier-Bessel en (r,θ), *i.e.* quand

$$f_{m,n,\omega}(r,v)=J_m(\kappa r)\times g(v)$$

et $\Phi_{m,n,\omega}(r)=J_m(\kappa r)$ (voir la proposition dans la section 3). En général, nous ne sommes pas dans ce cas et cela explique les différences que nous observons sur la Fig. 8.10. Ainsi, en considérant les précédentes approximations, on obtient :

$$-\left(\frac{\partial_r^2\phi_{m,n,\omega}}{\phi_{m,n,\omega}}+\left(\frac{1}{r}+\frac{\partial_r n_0(r)}{n_0(r)}\right)\frac{\partial_r\phi_{m,n,\omega}}{\phi_{m,n,\omega}}-\frac{m^2}{r^2}\right)+\frac{1}{T_e(r)}=J_0(\kappa\sqrt{2\mu})^2\frac{1}{n_0(r)}\int\frac{\frac{m}{r}\partial_r f_0+k\partial_v f_0}{kv-\omega}dv.$$

En posant

$$I=\int\frac{\frac{m}{r}\partial_r f_0+k\partial_v f_0}{kv-\omega}dv$$

et en utilisant l'expression de f_0, nous obtenons

$$I=\int\frac{-\frac{v}{T_i}+\frac{m}{kr}\left(\frac{\partial_r n_0}{n_0}-\frac{\partial_r T_i}{2T_i}+\frac{v^2\partial_r T_i}{2T_i^2}\right)}{v-\frac{\omega}{k}}f_0 dv.$$

Maintenant, nous introduisons pour $n \in \mathbb{N}$:

$$I_n = \frac{1}{n_0} \int v^n \frac{f_0}{v - \frac{\omega}{k}} f_0 dv$$

et nous obtenons les relations :

$$I_1 = 1 + \frac{\omega}{k} I_0, \qquad I_2 = \frac{\omega}{k} \left(1 + \frac{\omega}{k} I_0\right).$$

en utilisant le changement de variables $v = (2T_i(r))^{1/2} w$ et l'expression de f_0, nous avons en posant $k^* = (2T_i)^{1/2} k$:

$$\begin{aligned} I_0 &= \int \frac{\exp\left(-\frac{v^2}{2T_i}\right)}{(2\pi T_i)^{1/2}(v - \frac{\omega}{k}) dv} \\ &= \int \frac{\exp(-\omega)}{\pi^{1/2}((2T_i(r))^{1/2} w - \frac{\omega}{k}) dv} \\ &= \frac{1}{(2T_i)^{1/2}} Z\left(\frac{\omega}{k^*}\right) \end{aligned}$$

avec

$$Z(z) = \frac{1}{\sqrt{\pi}} \int \frac{\exp(-x^2)}{x - z} dx = i\sqrt{\pi} \exp(-z^2)(1 - \mathrm{erf}(-iz)),$$

$$\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x \exp(-t^2) dt.$$

Finalement, pour $z = \omega / k^*$:

$$\frac{J_0(\kappa\sqrt{2\mu})^2}{n_0(r)} I = -\frac{1}{T_i}(1 + zZ(z)) + \frac{m}{k^* r}\left(Z(z)\left(\frac{\partial_r n_0}{n_0} - \frac{\partial_r T_i}{2T_i}\right) + z(1 + zZ(z))\frac{\partial_r T_i}{T_i}\right). \tag{G.3}$$

Bibliographie

Chapitre 1

[1] C.Z. Cheng, G. Knorr, *The integration of the Vlasov equation in conguration space*, Journal of Computational Physics **22** (1976), pp. 330–351.

[2] N. Crouseilles, E. Faou, M. Mehrenberger, *High order Runge-Kutta-Nyström splitting methods for the Vlasov-Poisson equation*, inria-00633934, version 1. (2011)

[3] Y. Güçlü, A.J. Christlieb, W.N.G. Hitchon, *Arbitrarily high order Convected Scheme solution of the Vlasov-Poisson system*, Journal of Computational Physics **270** (2014), pp. 711–752.

[4] M. Shoucri, *Eulerian codes for the numerical solution of the Vlasov equation*, Commun. Nonlinear Sci. Numer. Simul. **1** (2008), pp. 174–182.

[5] E. Sonnendrücker, *Numerical methods for the Vlasov equations*, Lecture Notes, Max-Planck-Institut für Plamaphysik. (2013)

[6] E. Sonnendrücker, J. Roche, P. Bertrand, A. Ghizzo, *The semi-Lagrangian method for the numerical resolution of the Vlasov equation*, Journal of Computational Physics, **149** (1999), pp. 201–220.

Chapitre 2

[7] R.W. Beam, R.F. Warming, *An implicit finite difference algorithm for hyperbolic systems in conservation form*, J. Comput. Phys. **23** (1976), pp. 87–110.

[8] D. Bouche, G. Bonnaud, *Comparaisons de schémas numériques résolvant l'équation d'advection*, Rapport CEA-R-5967 (2001).

[9] D. Bouche, G. Bonnaud, D. Ramos, *Comparison of Numerical Schemes for Solving the Advection Equation*, Applied Mathematics Letters, **16** (2003), pp. 147–154.

[10] N. Crouseilles, M. Mehrenberger, E. Sonnendrücker, *Conservative semi-Lagrangian schemes for Vlasov equations*, J. Comput. Phys. **229** (2010), pp. 1927–1953.

[11] B. Després, *Uniform asymptotic stability of Strang's explicit compact schemes for linear advection*, SIAM **47**, No. 5 (2009), pp. 3956–3976.

[12] E. Franck, *Construction et analyse numérique de schéma asymptotic preserving sur maillages non structurés. Application au transport linéaire et aux systèmes de Friedrichs*, Thèse, Université Pierre et Marie Curie - Paris VI (2012).

[13] A. Iserles, G. Strang, *The optimal accuracy of difference schemes*, Trans. Amer. Math. Soc. **277** (1983), pp. 779–803.

[14] D. Kershaw, *The Explicit Inverses of Two Commonly Occurring Matrices*. Mathematics of Computation **23**, No. 105 (1969), pp. 189–19.

[15] P.D. Lax, B. Wendroff, *Systems of conservation laws*, C.P.A.M. **13** (1960), pp. 217–237.

[16] Y. Shokin, *The method of differential approximation*, Springer, Berlin (1983).

[17] G. Strang, *Trigonometric polynomials and difference methods of maximum accuracy*, J. Math. Phys. **41** (1962), pp. 147–154.

Chapitre 3

[18] N. Besse, *Convergence of a semi-Lagrangian scheme for the one-dimensional Vlasov-Poisson system,* SIAM, J. Numer. Anal., **42** (2004), pp. 350–382.

[19] N. Besse, *Convergence for a high-order semi-Lagrangian scheme with propagation of gradients for the Vlasov-Poisson system,* SIAM, J. Numer. Anal., **46** (2008), pp. 639–670.

[20] M. Bostan, *The Vlasov-Poisson system with strong external magnetic field. Finite Larmor radius regime,* Asymptot. Anal., **61**, No. 2 (2009), pp. 91–123.

[21] O. Bokanowski, Y. Cheng, C.W. Shu, *Convergence of discontinuous Galerkin schemes for front propagation with obstacles*, hal-00834342, (2012).

[22] O. Bokanowski, G. Simarmata, *Semi-Lagrangian discontinuous Galerkin schemes for some first and second order partial differential equations*, hal-00743042, (2012).

[23] F. Charles, B. Després, M. Mehrenberger, *Enhanced convergence estimates for semi-lagrangian schemes Application to the Vlasov-Poisson equation,* SIAM J. Numer. Anal. **51** (2013), pp. 840–863.

[24] B. Cockburn, M. Luskin, C.W. Shu, E. Suli, *Enhanced accuracy by post-processing for finite element methods for hyperbolic equations,* Mathematics of Computation **72** (2003), pp. 577–606.

[25] N. Crouseilles, M. Mehrenberger, F. Vecil, *Discontinuous Galerkin Semi-Lagrangian Method for Vlasov-Poisson*, CEMRACS'10 research achievements : Numerical modeling of fusion, ESAIM Proceedings **32** (2011), pp. 211–230.

[26] W. Guo, X. Zhong, J-M. Qiu, *Superconvergence of discontinuous Galerkin and local discontinuous Galerkin methods : eigen-structure analysis based on Fourier approach*, Journal of Computational Physics **235** (2013), pp. 458–485.

[27] A. Mangeney, F. Califano, C. Cavazzoni, P. Travnicek, *A Numerical Scheme for the Integration of the Vlasov-Maxwell System of Equations,* Journal of Computational Physics **179**, No. 2 (2002), pp. 495–538.

[28] J-M. Qiu, C. W. Shu, *Positivity preserving semi-Lagrangian discontinuous Galerkin formulation : Theoretical analysis and application to the Vlasov-Poisson system,* Journal of Computational Physics **230**, No. 23 (2011), pp. 8386–8409.

[29] M. Restelli, L., Bonaventura, R. Sacco, *A semi-Lagrangian discontinuous Galerkin method for scalar advection by incompressible flows,* Journal of Computational Physics **216**, No. 1 (2006), pp. 195–215.

[30] J.A. Rossmanith, D.C. Seal, *A positivity-preserving high-order semi-Lagrangian discontinuous Galerkin scheme for the Vlasov-Poisson equations,* Journal of Computational Physics **230**, No. 16 (2011), pp. 6203–6232.

Chapitre 4

[31] T. D. Arber, R. G. Vann, *A critical comparison of Eulerian-grid-based Vlasov solvers*, J. Comput. Phys. **180** (2002), pp. 339–357.

[32] M. Baldauf, *Stability analysis for linear discretisations of the advection equation with Runge-Kutta time integration*, J. Comput. Phys. **227** (2008), pp. 6638–6659.

[33] J. W. Banks, J. A. F. Hittinger, *A new class of nonlinear finite-volume methods for Vlasov simulation*, IEEE Trans. Plasma Sc **38** (2010).

[34] A. S. Bonnet-Bendhia, S. Fliss, P. Joly, P. Moireau, *Introduction aux équations aux dérivées partielles et à leur approximation numérique*, polycopié, cours ENSTA (2011).

[35] J. P. Boris, D. L. Book, *Flux-corrected transport. I : SHASTA, a fluid transport algorithm that works*, J. Comput. Phys. **11** (1973), pp. 38–69.

[36] P. Colella, P. R. Woodward, *The piecewise parabolic method (PPM) for gas-dynamical simulations*, J. Comput. Phys. **54** (1984), pp. 174–201.

[37] P. Colella, M. R. Dorr, J. A. F. Hittinger, D. F. Martin, *High-Order, Finite-Volume Methods in Mapped Coordinates*, J. Comput. Phys. **230** (2011), pp. 2952–2976.

[38] G. Dimarco, L. Pareschi, *Numerical Methods for Kinetic Equations,* Acta Numerica **23** (2014), pp. 369–520.

[39] G. Dimarco, R. Loubere, *Towards an ultra efficient kinetic scheme. Part I : Basics on the BGK equation,* Journal of Computational Physics **255** (2013), pp. 680–698.

[40] G. Dimarco, R. Loubere, *Towards an ultra efficient kinetic scheme. Part II : The high order case,* Journal of Computational Physics **255** (2013), pp. 699–719.

[41] N. Elkina, J. Büchner, *A new conservative unsplit method for the solution of the Vlasov equation,* J. Comput. Phys. **213** (2006), pp. 862–875.

[42] E. Fijalkow, *A numerical solution to the Vlasov equation*, Comput. Phys. Commun. **116** (1999), pp. 329–335.

[43] F. Filbet, E. Sonnendrücker, P. Bertrand, *Conservative numerical schemes for the Vlasov equation*, J. Comput. Phys. **172** (2001), pp. 166–187.

[44] F. Filbet, E. Sonnendrücker, *Comparison of Eulerian Vlasov solvers*, Comput. Phys. Comm. **151** (2003), pp. 247–266.

[45] F. Huot, A. Ghizzo, P. Bertrand, E. Sonnendrücker, O. Coulaud, *Instability of the time splitting scheme for the one-dimensional and relativistic Vlasov-Maxwell system*, J. Comput. Phys. **185** (2003), pp. 512–531.

[46] P.H. Lauritzen, R. D. Nair, P. A. Ullrich, *A conservative semi-Lagrangian multi-tracer transport scheme (CSLAM) on the cubed-sphere grid*, J. Comput. Phys. **229** (2010), pp. 1401–1424.

[47] J.M. Qiu, C. W. Shu, *Conservative semi-Lagrangian finite difference WENO formulations with applications to the Vlasov equation*, Comm. Comput. Phys. **10** (2011), pp. 979–1000.

[48] T. Schwartzkopff, M. Dumbser, C.D. Munz, *Fast high order ADER schemes for linear hyperbolic equations and their numerical dispersion and dissipation,* J. Comput. Phys. **197** (2004), pp. 532–538.

[49] M. Shoucri, *Nonlinear evolution of the bump-on-tail instability*, Phys. Fluids **22** (1979), pp. 2038–2039.

[50] P. Glanc, *Approximation numérique de l'équation de Vlasov par des méthodes de type remapping conservatif*, Thèse, Université de Strasbourg. (2014)

Chapitre 5

[51] B. Afeyan, K. Won, V. Savchenko, T. Johnston, A. Ghizzo, P. Bertrand. *Kinetic Electrostatic Electron Nonlinear (KEEN) Waves and their Interactions Driven by the Ponderomotive Force of Crossing Laser Beams.*, Proc. IFSA 2003, p. 213 and arXiv :1210.8105, http://arxiv.org/abs/1210.8105.

[52] N. Besse, M. Mehrenberger, *Convergence of classes of high-order semi-lagrangian schemes for the Vlasov-Poisson system,* Mathematics of Computation **77** (2008), pp. 93–123.

[53] C. K. Birdsall, A. B. Langdon, *Plasma Physics via Computer Simulation*, Adam Hilger (1991).

[54] K. J. Bowers, B. J. Albright, B. Bergen, L. Yin, K. J. Barker, D. J. Kerbyson, 0.374 *pflop/s trillion-particle kinetic modeling of laser plasma interaction on roadrunner*, Proc. of Supercomputing. IEEE Press (2008).

[55] Y. Cheng, I. M. Gamba, P. J. Morrison, *Study of conservation and recurrence of Runge-Kutta discontinuous Galerkin schemes for Vlasov-Poisson systems*, J. Sci. Comput. **56**, No. 2 (2013), pp. 319–349.

[56] A. Crestetto, P. Helluy, *Resolution of the Vlasov-Maxwell system by PIC Discontinuous Galerkin method on GPU with OpenCL*, ESAIM Proc. 2011.

[57] T. Dannert, *GENE on Accelerators*, 4th Summer school on numerical modeling for fusion, 8-12 October 2012, IPP, Garching near Munich, Germany, http://www.ipp.mpg.de/ippcms/eng/for/veranstaltungen/konferenzen/su_school/.

[58] F. Filbet *Numerical simulations avalaible online at* http://math.univ-lyon1.fr/~filbet/publication.html.

[59] R.M. Gray, *Toeplitz and circulant matrices : a review*, Now Publishers Inc, Boston-Delft (2005).

[60] Y. Guclu, W. N. G. Hitchon, Szu-Yi Chen, *High order semi-lagrangian methods for the kinetic description of plasmas,* Plasma Science (ICOPS), 2012 Abstracts IEEE, 8-13 July 2012, doi : 10.1109/PLASMA.2012.6383976.

[61] R. Hatzky, *Global electromagnetic gyrokinetic particle-in-cell simulation*, 4th Summer school on numerical modelling for fusion, 8-12 October 2012, IPP, Garching near Munich, Germany, http://www.ipp.mpg.de/ippcms/eng/for/veranstaltungen/konferenzen/su_school/.

[62] N.A. Krall, A.W. Trivelpiece, *Principles of Plasma Physics*, McGraw–Hill, New York (1973).

[63] G. Latu, *Fine-grained parallelization of Vlasov-Poisson application on GPU*, Euro-Par 2010, Parallel Processing Workshops, Springer (New York, 2011).

[64] G. Manfredi, *Long time behavior of nonlinear Landau damping,* Physical review letters **79**, No. 15 (1997), pp. 2815–2818.

[65] C. Mouhot, C. Villani, *On Landau damping*, Acta Mathematica **207**, No. 1 (2011), pp. 29–201. http://arxiv.org/abs/0904.2760.

[66] T. M. Rocha Filho, *Solving the Vlasov equation for one-dimensional models with long range interactions on a GPU*, Computer Physics Communications **184**, No. 1 (2013), pp. 34–39.

[67] *Selalib, a semi-Lagrangian library,* http://selalib.gforge.inria.fr/

[68] E. Sonnendrücker , N. Crouseilles , B. Afeyan, *BP8.00057 : High Order Vlasov Solvers for the Simulation of KEEN Wave Including the L-B and F-P Collision Models*, 54th Annual Meeting of the APS Division of Plasma Physics Volume 57, Number 12, Monday–Friday, October 29–November 2 2012 ; Providence, Rhode Island, http://meeting.aps.org/Meeting/DPP12/SessionIndex2/?SessionEventID=181483.

[69] G. Stantchev, W. Dorland, N. Gumerov, *Fast parallel particle-to-grid interpolation for plasma PIC simulations on the GPU*, J. Parallel Distrib. Comput. **68**, No. 10 (2008), pp. 1339–1349.

[70] T. Zhou, Y. Guo, C.W. Shu, *Numerical study on Landau damping*, Physica D **157** (2001), pp. 322–333.

Chapitre 6

[71] B. Afeyan, F. Casas, N. Crouseilles, A. Dodhy, E. Faou, M. Mehrenberger, E. Sonnendrücker, *Simulations of Kinetic Electrostatic Electron Nonlinear (KEEN) Waves with Two-Grid, Variable Velocity Resolution and High-Order Time-Splitting*, accepté à Eur. Phys. J. D **68** (2014), p. 295.
DOI : 10.1140/epjd/e2014-50212-6

[72] L. Einkemmer, A. Ostermann, *Convergence analysis of Strang splitting for Vlasov-type equations.* SIAM Journal on Numerical Analysis **52**, No. 1 (2014), pp. 140–155.

[73] L. Einkemmer, A. Ostermann, *Convergence analysis of a discontinuous Galerkin/Strang splitting approximation for the Vlasov-Poisson equation.* SIAM Journal on Numerical Analysis **52**, No. 2 (2014), pp. 757–778

[74] R.E. Heath, I.M. Gamba, P.J. Morrison, C. Michler, *A discontinuous Galerkin method for the Vlasov-Poisson system,* Journal of Computational Physics **231**, No. 4 (2012), pp. 1140–1174.

[75] M. Mehrenberger, C. Steiner, L. Marradi, N. Crouseilles, E. Sonnendrücker & B. Afeyan, *Vlasov on GPU*, ESAIM Proc. 2013.

Chapitre 7

[76] J.-P. Braeunig, N. Crouseilles, V. Grandgirard, G. Latu, M. Mehrenberger, E. Sonnendrücker *Some numerical aspects of the conservative PSM scheme in a 4D drift-kinetic code*. Rapport de recherche INRIA (2009).

[77] N. Crouseilles, P. Glanc, S. Hirstoaga, E. Madaule, M. Mehrenberger, J. Pétri *Semi-Lagrangian simulations on polar grids : from diocotron instability to ITG turbulence*, accepté à Eur. Phys. J. D **68** (2014), p. 252.
DOI : 10.1140/epjd/e2014-50180-9

[78] F. Filbet, C. Yang *Conservative and non-conservative methods based on Hermite weighted essentially-non-oscillatory reconstruction for Vlasov equations*, J. Comput. Physics, **279** (2014).

[79] F. Filbet, C. Yang *Mixed semi-Lagrangian/finite difference methods for plasma simulations*, soumis.

[80] V. Grandgirard, M. Brunetti, P. Bertrand, N. Besse, X. Garbet, P. Ghendrih, G. Manfredi, Y. Sarazin, O. Sauter, E. Sonnendrücker, J. Vaclavik, L. Villard *A drift-kinetic Semi-Lagrangian 4D code for ion turbulence simulation.* J. Comput. Physics, **217**, No. 2 (2006), pp. 395–423.

[81] R. Klein, E. Gravier, P. Morel, N. Besse, P. Bertrand *Gyrokinetic water-bag modeling of a plasma column : Magnetic moment distribution and finite Larmor radius effects*. Physics of plasmas **16**, 082106 (2009).

[82] J.M. De Villiers, C.H. Rohwer, *Optimal local spline interpolants*. Journal of Computational and Applied Mathematics **18**, No. 1 (1987), pp. 107–119.

Chapitre 8

[83] M. Abramowitz, I. A. Stegun, *Handbook of Mathematical Functions*, (Dover Publications, New York, 1965).

[84] M. Bostan, *Transport e,quations with disparate advection fields. Application to the gyrokinetic models in plasmas physics*, J. Differential Equations **249** (2010) pp. 1620–1663.

[85] A. Brizard, *Nonlinear gyrokinetic Maxwell-Vlasov equations using magnetic coordinates* J. Plasma Phys. **41** (1989), pp. 541–559.

[86] D. Coulette, N. Besse *Numerical comparisons of gyrokinetic multi-water-bag models.* JCP **248** (2013), pp. 1–32.

[87] N. Crouseilles, M. Mehrenberger, H. Sellama, *Numerical solution of the gyroaverage operator for the finite gyroradius guiding-center model*, CiCP **8** (2010), pp. 484–510.

[88] A. Dimits et al., *Comparisons and physics basis of tokamak transport models and turbulence simulations*, Physics of Plasmas **7**, No. 3 (2000), pp. 969–983.

[89] X. Garbet, Y. Idomura, L. Villard, T.H. Watanabe, *Gyrokinetic simulations of turbulent transport*, Nuclear Fusion **50**, No. 4 (2010), 043002.

[90] V. Grandgirard, Y. Sarazin, X. Garbet, G. Dif-Pradalier, Ph. Ghendrih, N. Crouseilles, G. Latu, E. Sonnendrücker, N. Besse, P. Bertrand, *Computing ITG turbulence with a full-f semi-Lagrangian code,* Communications in Nonlinear Science and Numerical Simulation **13** No. 1 (2008), pp. 81–87.

[91] T. Görler, *Multiscale effects in plasma microturbulence*, Thèse, Ulm (2009).

[92] T. Görler, X. Lapillonne, S. Brunner, T. Dannert, F. Jenko, F. Merz, D. Told, *The global version of the gyrokinetic turbulence code GENE.*, J. Comput. Physics **230**, No. 18 (2011), pp. 7053–7071.

[93] R. Hatzky, T.M. Tran, A. Konies, R. Kleiber, S.J. Allfrey, *Energy conservation in a nonlinear gyrokinetic particle-in-cell code for ion-temperature-gradient-driven modes in theta-pinch geometry*, Physics of Plasmas **9**, No. 3 (2002), pp. 898–912.

[94] Y. Idomura, S. Tokuda, Y. Kishimoto, M. Wakatani, *Gyrokinetic theory of drift waves in negative shear tokamaks*, Nuclear Fusion **41**, No. 4 (2001).

[95] S. Jolliet, A. Bottino, P. Angelino, R. Hatzky, T.M. Tran, B.F. Mcmillan, O. Sauter, K. Appert, Y. Idomura, L. Villard, *A global collisionless PIC code in magnetic coordinates*, Comp. Phys. Comm. **177**, No. 5 (2007), pp. 409–425.

[96] W. W. Lee, *Gyrokinetic approach in particle simulation*, Physics of Fluids **26**, No. 2 (1983), pp. 556–562.

[97] Y. Sarazin, V. Grandgirard, E. Fleurence, X. Garbet, Ph. Ghendrih, P. Bertrand, G. Depret, *Kinetic features of interchange turbulence*, Plasma Phys. Control. Fusion **47**, No. 10 (2005), pp. 1817–1840.

[98] M. Kreh, *Bessel Functions*, www.math.psu.edu/papikian/Kreh.pdf

Chapitre 9

[99] N. Besse, P. Bertrand *Gyro-water-bag approach in nonlinear gyrokinetic turbulence*, J. Comput. Phys. **228** (2009), pp. 3973–3995.

[100] N. Crouseilles, A. Ratnani, E. Sonnendrücker *An Isogeometric Analysis Approach for the study of the gyrokinetic quasi-neutrality equation*, Journal of Computational Physics **231**, No. 2 (2012), pp. 373–393.

[101] D.H.E. Dubin, J. A. Krommes, C. Oberman, W. W. Lee, *Nonlinear gyrokinetic equations*, Phys. Fluids **26**, No. 12 (1983).

[102] T.S. Hahm, *Nonlinear Gyrokinetic Equations for Tokamaks Microturbulence*, Phys. of Fluids **31** (1988).

[103] M. Hauray, A. Nouri, P. Ghendrih, *Derivation of a gyrokinetic model : existence and uniqueness of specific stationary solutions*, KRM **4** (2009), pp. 707–725.

[104] W.W. Lee, R.A. Kolesnikov, *On higher-order corrections to gyrokinetic Vlasov-Poisson equations in the long wavelength limit*, Phys. of Plasmas **16**, 044506 (2009).

[105] Z. Lin, W.W. Lee, *Method for solving the gyrokinetic Poisson equation in general geometry*, Phys. Rev. E **52** (1995), pp. 5646–5652.

[106] G. Manfredi, M. Shoucri, R.O. Dendy, A. Ghizzo, P. Bertrand, *Vlasov gyrokinetic simulations of ion-temperature-gradient driven instabilities*, Physics of Plasmas. **3**, No. 202 (1996), pp. 202–217.

[107] A. Mishchenko, A. Könies, R. Hatzky, *Particle simulations with a generalized gyrokinetic solver*, Phys. of Plasmas **12**, 062305 (2005).

[108] M. Shoucri, G. Manfredi, P. Bertrand, A. Ghizzo, J. Lebas, G. Knorr, *Charge-separation velocity shear and suppression of turbulence at a plasma edge in the gyrokinetic approximation*, Journal of Plasma Physics **61**, No. 2 (1999), pp. 191–212.

Christophe STEINER

Résolution numérique de l'opérateur de gyromoyenne, schémas d'advection et couplage. Applications à l'équation de Vlasov.

Résumé

Cette thèse propose et analyse des méthodes numériques pour la résolution de l'équation de Vlasov. Cette équation modélise l'évolution d'une espèce de particules chargées sous l'effet d'un champ électromagnétique. La première partie est consacrée à une analyse mathématique de schémas semi-Lagrangiens résolvant l'équation de transport linéaire qui constituent la brique de base des méthodes de splitting directionnel.

Des méthodes de résolution de l'équation de Vlasov couplée à l'équation de Poisson, dans le cas où uniquement le champ électrique est considéré, sont optimisées dans la seconde partie. Il s'agit d'optimisation en temps de calcul par l'utilisation de cartes graphiques (GPU) et l'utilisation d'un maillage non homogène.

Dans la troisième et dernière partie, nous étudierons une méthode numérique de calcul de l'opérateur de gyromoyenne intervenant dans la théorie gyrocinétique que nous appliquerons à l'équation de quasi-neutralité.

Mots-clés : Equation de Vlasov, Méthodes semi-Lagrangiennes, Equations équivalentes, Superconvergence, GPU, Modèle gyrocinétique, Gyromoyenne, Equation de quasi-neutralité

Abstract

This thesis proposes and analyzes numerical methods for solving the Vlasov equation. This equation models the evolution of a species of charged particles under the effet of an electromagnetic field. The first part is devoted to a mathematical analysis of semi-Lagrangian schemes solving the linear transport equation which is the basic building block of directional splitting methods.

Solving methods for the Vlasov equation coupled to the Poisson equation, in the case where only the electric field is considered, are optimized in the second part. This optimization relates to the time of calculation by the use of Graphics Processing Unit (GPU) and the use of an inhomogeneous mesh.

In the third and final part, we study a numerical method for calculating the gyroaverage operator involved in gyrokinetic theory. This method will be applied to solve the quasi-neutrality equation.

Keywords : Vlasov equation, Semi-Lagrangian methods, Equivalent equations, Superconvergence, GPU, Gyrokinetic model, Gyroaverage, Quasi-neutrality equation.